Free LPIC1 Course (Book & Videos)

Original with free videos: Linux1st.com

jadijadi@gmail.com

Know your LPIC1 version 5 exam

This document is written based on LPIC1 version 5 (exams 101-500 and 102-500). It covers all the objectives. Each LPIC exam has 60 multiple choice and fill-in-the-blank questions which should be answered in 90 minutes.

Prepare your Lab

Watching videos or even reading books is never enough. You need to have Linux installed and practice your newly acquired knowledge. As you may know, LPIC is a *distro agnostic* test and an LPIC-certified guru should be able to work in any GNU/Linux system, or at least on the major ones. Because of this, I suggest you have two Linux machines which you can work on. One RPM-based and one Debian-based.

One common way is to use a Virtual Machine. You can install something like VirtualBox (for Win/Linux and Intel Macs) or UTM (for M1 Macs) or any other solution to install your test machines in a Virtual Machine.

For your distribution, I suggest Ubuntu or Debian as your Debian-based system and Fedora or Rocky Linux as your RPM-based machine. Feel free to install anything else and do whatever you enjoy!

If you want to walk in my steps, do a normal GUI installation. Here is me, installing the Ubuntu 22.04:

And here, you can follow the Fedora 36 installation:

Note: The version is not that important. Here I'm installing Fedora 36 & Ubuntu 22.04.

101.1 Determine and configure hardware settings

Weight: 2

Candidates should be able to determine and configure fundamental system hardware.

Objectives

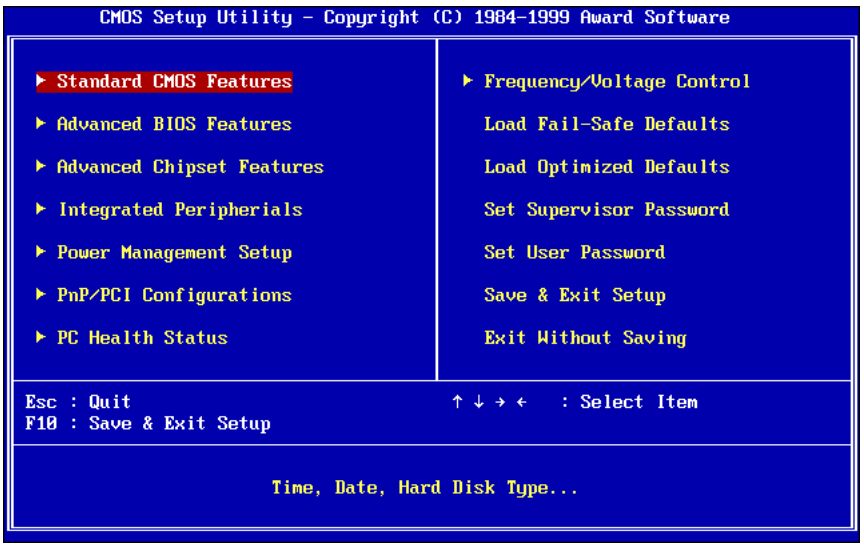
- Enable and disable integrated peripherals
- Differentiate between the various types of mass storage devices
- Determine hardware resources for devices
- Tools and utilities to list various hardware information (e.g. lsusb, lspci, etc.)
- Tools and utilities to manipulate USB devices
- Conceptual understanding of sysfs, udev, dbus
- `/sys`
- `/proc`
- `/dev`
- modprobe
- lsmod
- lspci
- lsusb

Find out about the hardware

An operating system (OS) is system software that manages computer hardware, and software resources, and provides common services for computer programs. It sits on top of the hardware and manages the resources when other software (Sometimes called an userspace program) asks for it.

Firmware is the software *on* your hardware that runs it; Think of it as a built-in OS or driver for your hardware. Motherboards need some firmware to be able to work too.

Firmware is a type of software that lives in hardware. Software is any program or group of programs run by a computer.



1. BIOS (Basic Input/Output System). Old and redundant. It is intractable through a text menu-based system and it boots the computer by accessing the first sector of the first partition of your hard disk (MBR). This is not enough for modern systems and most systems use a two-step boot procedure.

2. UEFI (Unified Extensible Firmware Interface). Started as EFI in 1998 in Intel. Now the standard. Uses a specific disk partition for boot (EFI System Partition (ESP)) and uses FAT. On Linux it's located on `/boot/efi` and the files use the `.efi` extension. You need to register each bootloader.

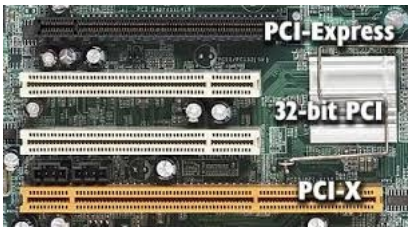
BIOS	UEFI
Windows blue screen	User-friendly graphical user interface
No Animation	Multi colored with Animations
Can't recognize Ethernet, Wi-Fi and Bluetooth	Support Ethernet, Wi-Fi and Bluetooth connectivity
No remote diagnosis and repair support	Support remote diagnosis and repair if the OS won't boot.
No mouse support, keyboard only	Keyboard and mouse support
No secure booting option	Secure boot facility to stops loading malicious softwares.
Firmware program in 16-bit assembly language	Firmware program in 64-bit C language
Supports 2.2 terabytes	Supports drive sizes upto 9 zettabytes

Peripheral Devices

These are device interfaces.

PCI

Peripheral Component Interconnect. Enables the user to add extra components to the Motherboard. Now most servers use PCI Express (PCIe)



- Internal HDD.
  - PATA (old)
  - SATA (serial & up to 4 devices) (II-III)
  - SCSI (parallel & up to 8 devices) (Small Computer System Interface)
- External HDD. Fiber (Providing a high-speed data connection). SSD over USB
- Network cards. RJ 45 (Registered Jack 45)
- Wireless cards. IEEE 802.11
- Bluetooth. IEEE 802.15 (Short-range (up to 10m) wireless technology standard)
- Video accelerators. Hardware circuits on a graphics card that speed up full-motion video.
- Audio cards

SSD vs HDD

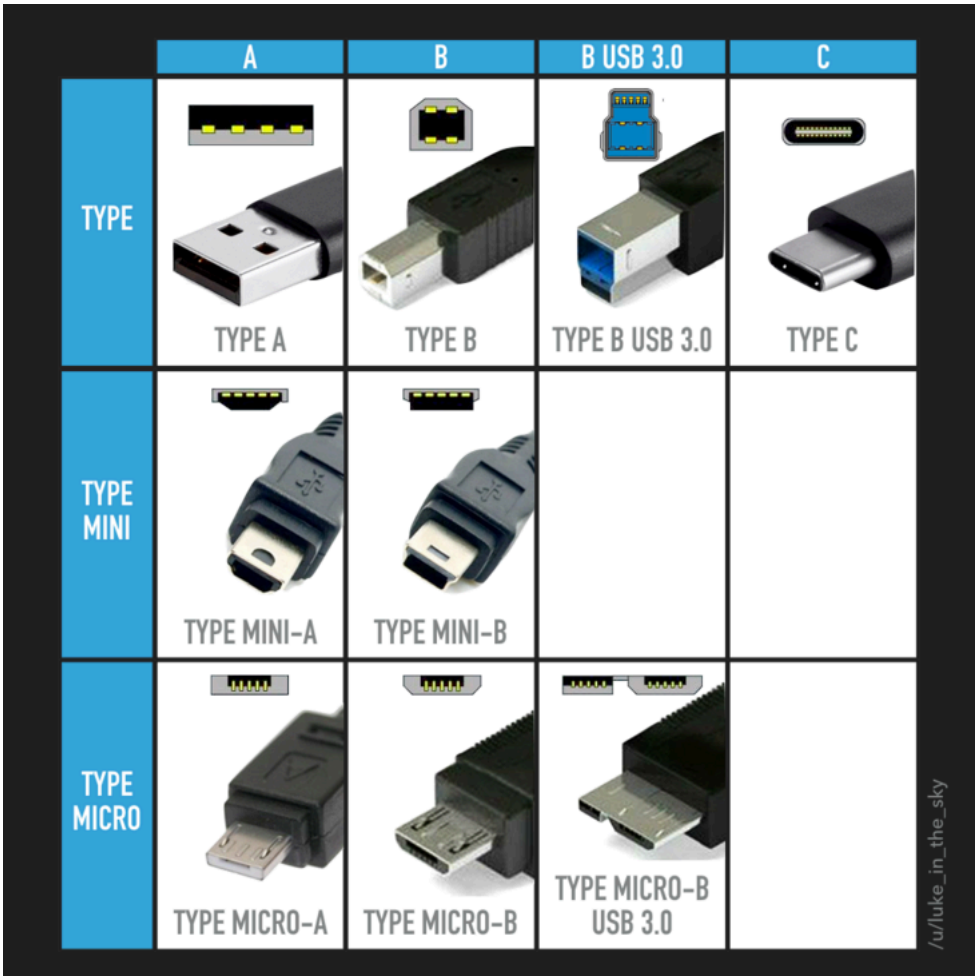
- SSDs are faster (reads up to 10 times and writes up to 20 times faster), quieter, smaller, more durable, and consume less energy, while HDDs are cheaper and offer more storage capacity and easier data recovery if damaged.
- SSDs don't have moving parts such as actuator arms and spinning platters like hard drives (an SSD uses flash memory without any moving parts). That's one reason why SSDs can withstand accidental drops and other shocks, vibration, extreme temperatures, and magnetic fields better than HDDs.
- HDDs tend to last around 3-5 years, SSDs can last up to 10 years or more. (Current SSDs have reserve capacities. These storage spaces aren't available to the user, but are used to repair damaged cells, so to speak. The defect cells are replaced with brand-new reserve cells; this procedure is called “Bad-Block-Management”. Thus, SSD storage cells in normal operation last a lifetime.)

Network cards vs Wireless cards

- NIC (Network Interface Card) provides computing network connectivity on a computing device to Ethernet network in Home or Office through RJ45 port. Wireless Adapter Card provides wireless connectivity to computing network enabled by Access Points on premises (in home or office) on a Computing Device.
- Wireless cards are installed on an industrial computer is used to enable wireless connectivity to the internet.

USB

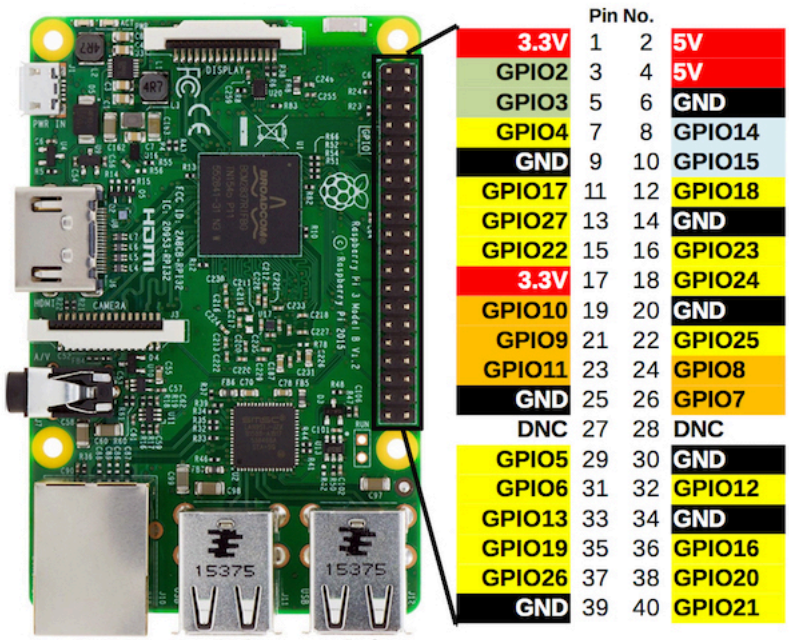
Universal Serial Bus. Serial and need fewer connections.



- 1 (12Mbps), 2 (480Mbps), 3 (PCI.e 2.0 Bus:5Gbps, PCI.e 3.0 Bus:10Gbps, PCI.e 3.2 Bus:20Gbps, PCI.e 4.0 Bus:40Gbps)
- A, B, C

GPIO

General Purpose Input Output.



- To control other devices
- Examples include Arduino, raspberrypi, etc.

Sysfs

Sysfs is a pseudo file system provided by the Linux kernel that exports information about various kernel subsystems, hardware devices, and associated device drivers from the kernel's device model to user space through virtual files.[1] In addition to providing information about various devices and kernel subsystems, exported virtual files are also used for their configuration.

Sysfs is mounted under the `/sys` mount point.

Copy

```
jadi@funlife:~$ ls /sys
block  bus  class  dev  devices  firmware  fs  hypervisor  kernel  module  power
```

All block devices are at the `block` and `bus` directory has all the connected PCI, USB, serial, ... devices. Note that here in `sys` we have the devices based on their technology but `/dev/` is abstracted.

udev

udev (userspace `/dev`) is a device manager for the Linux kernel. As the successor of devfsd and hotplug, udev primarily manages device nodes in the `/dev` directory. At the same time, udev also handles all user space events raised when hardware devices are added into the system or removed from it, including firmware loading as required by certain devices.

There are a lot of devices in `/dev/` and if you plug in any device, it will be assigned a file in `/dev` (say `/dev/sdb2`). **udev** lets you control what will be what in `/dev`. For example, you can use a rule to force your 128GB flash drive with one specific vendor to be `/dev/mybackup` every single time you

connect it and you can even start a backprocess as soon as it connects.

In essence, **udev** serves as the custodian of the `/dev/` directory. It abstracts the representation of devices, such as a hard disk, which is identified as `/dev/sda` or `/dev/hd0`, irrespective of its manufacturer, model, or underlying technology.

Copy

```
root@funlife:/dev# ls /dev/sda*
/dev/sda  /dev/sda1  /dev/sda2  /dev/sda3  /dev/sda5  /dev/sda6
```

If a program wants to read/write from/to a device, it will use the corresponding file in /dev to do so. This can be done on **character devices** or **block devices**. When listing, a `b` or `c` will indicate this:

Copy

```
root@ocean:~# ls -ltrh /dev/  # Partial output is shown
crw-rw---- 1 root tty       4,   1 Dec 15  2019 tty1
crw-rw-rw- 1 root root      1,   5 Dec 15  2019 zero
brw-rw---- 1 root disk      1,   0 Dec 15  2019 ram0
brw-rw---- 1 root disk 253,   0 Dec 15  2019 /dev/vda
```

dbus

D-Bus is a message bus system, a simple way for applications to talk to one another. In addition to inter-process communication, D-Bus helps coordinate process lifecycle; It makes it simple and reliable to code a "single instance" application or daemon and to launch applications and daemons on demand when their services are needed.

proc directory

This is where the Kernel keeps its settings and properties. This directory is created on ram and files might have write access (say for some hardware configurations). You can find things like:

- IRQs (interrupt requests)
- I/O ports (locations in memory where CPU can talk with devices)
- DMA (direct memory access, faster than I/O ports)
- Processes
- Network Settings
- ...

Copy

```
$ ls /proc/
1      1249   1451   1565   18069  20346  2426  2765  2926  3175  3317  3537  39    468   4921  53    689   969      filesystems  misc       s
10     13     146    157    18093  20681  2452  2766  2929  3183  3318  354   397   4694  4934  538   7     97       fs       modules    t
1039   1321   147    1572   18243  21     2456  28    2934  3187  34    3541  404   4695  4955  54    737   acpi     interrupts   mounts
10899  13346  148    1576   18274  21021  2462  2841  2936  3191  3450  3550  41    47    4970  546   74    asound   iomem        mtrr       t
10960  13438  14817  158    1859   21139  25    2851  2945  32    3459  357   42    4720  4982  55    742   buddyinfo    ioports      net
11     13619  149    16     18617  2129   2592  2852  2947  3202  3466  36    43    4731  4995  551   75    bus      irq          pagetypeinfo
11120  13661  15     1613   18781  214    26    2862  2948  3206  3467  3683  44    4756  5 56    77    cgroups  kallsyms     partitions     v
11145  13671  150    1630   1880   215    27    2865  2952  3208  3469  3699  4484  4774  50    577   8     cmdline  kcore        sched_debug
1159   13927  151    1633   1882   2199   2707  2866  2955  3212  3470  37    4495  4795  5008  5806  892   consoles     keys         schedsta
1163   14     1512   1634   19     22     2708  2884  2957  3225  3474  3710  45    48    5013  60    9     cpuinfo  key-users    scsi       z
1164   14045  1515   1693   19061  2219   2709  2887  2961  3236  3475  3752  4506  4811  5077  61    904   crypto   kmsg         self
1170   14047  152    17     19068  23     2710  2891  3     324   3477  3761  4529  4821  5082  62    9061  devices  kpagecount   slabinfo
1174   14052  153    17173  19069  23055  2711  2895  3047  3261  3517  3778  4558  484   5091  677   915   diskstats    kpageflags   softirqs
12     1409   154    1732   19075  2354   2718  29    3093  3284  3522  38    4562  4861  51    678   923   dma      loadavg      stat
1231   1444   155    17413  2      2390   2719  2904  31    3287  3525  3803  46    4891  52    679   939   driver   locks        swaps
1234   1446   156    17751  20     24     2723  2908  3132  3298  3528  3823  4622  49    5202  680   940   execdomains  mdstat       sys
1236   145    1563   18     2028   2418   2763  2911  3171  33    3533  3845  4661  4907  525   687   96    fb       meminfo      sysrq-trigge
```

The numbers are the process IDs! There are also other files like `cpuinfo`, `mounts`, `meminfo`, ...

Copy

```
$ cat /proc/cpuinfo
processor   : 0
vendor_id   : GenuineIntel
cpu family  : 6
model       : 42
model name  : Intel(R) Core(TM) i5-2520M CPU @ 2.50GHz
stepping    : 7
microcode   : 0x15
cpu MHz     : 3195.312
cache size  : 3072 KB
physical id : 0
siblings    : 4
```

```
core id     : 0
cpu cores   : 2
apicid      : 0
initial apicid  : 0
fpu     : yes
fpu_exception   : yes
cpuid level : 13
wp      : yes
flags       : fpu vme de pse tsc msr pae mce cx8 apic sep mtrr pge mca cmov pat pse36 clflush dts acpi mmx fxsr sse sse2 ss ht tm pbe syscall
bogomips    : 4983.79
clflush size    : 64
cache_alignment : 64
address sizes   : 36 bits physical, 48 bits virtual
power management:

processor   : 1
vendor_id   : GenuineIntel
cpu family  : 6
model       : 42
model name  : Intel(R) Core(TM) i5-2520M CPU @ 2.50GHz
stepping    : 7
microcode   : 0x15
cpu MHz     : 3010.839
cache size  : 3072 KB
physical id : 0
siblings    : 4
core id     : 0
cpu cores   : 2
apicid      : 1
initial apicid  : 1
fpu     : yes
fpu_exception   : yes
cpuid level : 13
wp      : yes
flags       : fpu vme de pse tsc msr pae mce cx8 apic sep mtrr pge mca cmov pat pse36 clflush dts acpi mmx fxsr sse sse2 ss ht tm pbe syscall
bogomips    : 4983.79
clflush size    : 64
cache_alignment : 64
address sizes   : 36 bits physical, 48 bits virtual
power management:
```

We can also write here. Since I'm on an IBM Lenovo laptop I can turn my LED on and off by writing here:

Copy

```
root@funlife:/proc/acpi/ibm# echo on > light
root@funlife:/proc/acpi/ibm# echo off > light
```

One more traditional example is changing the max number of open files per user:

Copy

```
root@funlife:/proc/sys/fs# cat file-max
797946
root@funlife:/proc/sys/fs# echo 1000000 > file-max
root@funlife:/proc/sys/fs# cat file-max
1000000
```

Another very useful directory here, is `/proc/sys/net/ipv4` which controls real-time networking configurations.

> All these changes will be reverted after a boot. You have to write into config files in `/etc/` to make these changes permanent

Try yourself! Check the `/proc/ioports` or `/proc/dma` or `/proc/iomem`.

lsusb, lspci, lsblk, lshw

Just like `ls` but for pci, usb, ...

lspci

Shows PCI devices that are connected to the computer.

Copy

```
# lspci
00:00.0 Host bridge: Intel Corporation 2nd Generation Core Processor Family DRAM Controller (rev 09)
00:02.0 VGA compatible controller: Intel Corporation 2nd Generation Core Processor Family Integrated Graphics Controller (rev 09)
00:16.0 Communication controller: Intel Corporation 6 Series/C200 Series Chipset Family MEI Controller #1 (rev 04)
00:19.0 Ethernet controller: Intel Corporation 82579LM Gigabit Network Connection (rev 04)
00:1a.0 USB controller: Intel Corporation 6 Series/C200 Series Chipset Family USB Enhanced Host Controller #2 (rev 04)
00:1b.0 Audio device: Intel Corporation 6 Series/C200 Series Chipset Family High Definition Audio Controller (rev 04)
00:1c.0 PCI bridge: Intel Corporation 6 Series/C200 Series Chipset Family PCI Express Root Port 1 (rev b4)
00:1c.1 PCI bridge: Intel Corporation 6 Series/C200 Series Chipset Family PCI Express Root Port 2 (rev b4)
00:1c.4 PCI bridge: Intel Corporation 6 Series/C200 Series Chipset Family PCI Express Root Port 5 (rev b4)
00:1d.0 USB controller: Intel Corporation 6 Series/C200 Series Chipset Family USB Enhanced Host Controller #1 (rev 04)
00:1f.0 ISA bridge: Intel Corporation QM67 Express Chipset Family LPC Controller (rev 04)
00:1f.2 SATA controller: Intel Corporation 6 Series/C200 Series Chipset Family 6 port SATA AHCI Controller (rev 04)
00:1f.3 SMBus: Intel Corporation 6 Series/C200 Series Chipset Family SMBus Controller (rev 04)
03:00.0 Network controller: Intel Corporation Centrino Wireless-N 1000 [Condor Peak]
0d:00.0 System peripheral: Ricoh Co Ltd MMC/SD Host Controller (rev 07)
```

lsusb

Shows all the USB devices connected to the system.

Copy

```
# lsusb
Bus 002 Device 003: ID 1c4f:0026 SiGma Micro Keyboard
Bus 002 Device 002: ID 8087:0024 Intel Corp. Integrated Rate Matching Hub
Bus 002 Device 001: ID 1d6b:0002 Linux Foundation 2.0 root hub
Bus 001 Device 005: ID 04f2:b217 Chicony Electronics Co., Ltd Lenovo Integrated Camera (0.3MP)
Bus 001 Device 004: ID 0a5c:217f Broadcom Corp. BCM2045B (BDC-2.1)
Bus 001 Device 003: ID 192f:0916 Avago Technologies, Pte.
Bus 001 Device 002: ID 8087:0024 Intel Corp. Integrated Rate Matching Hub
Bus 001 Device 001: ID 1d6b:0002 Linux Foundation 2.0 root hub
```

lshw

Shows hardware. Might need root status to get the full list. Test it!

lsblk

Used for list devices that can read from or write to by blocks of data.

Loadable Kernel Modules

Linux like any other OS needs drivers to work with hardware. In Microsoft Windows, you need to install the drivers separately but in Linux, the system has most of the drivers built-in. But to prevent the kernel from loading all of them at the same time and to decrease the Kernel size, Linux uses Kernel Modules. Loadable kernel modules (.ko files) are object files that are used to extend the kernel of the Linux Distribution. They are used to provide drivers for new hardware like IoT expansion cards that have not been included in the Linux Distribution.

You can inspect the modules using the `lsmod` or manage them via `modprobe` commands.

lsmod

Shows kernel modules. They are located at `/lib/modules`.

Copy

```
root@funlife:/dev# lsmod
Module                  Size  Used by
pci_stub               12622  1
vboxpci                23256  0
vboxnetadp             25670  0
vboxnetflt             27605  0
vboxdrv               418013  3 vboxnetadp,vboxnetflt,vboxpci
ctr                    13049  3
ccm                    17731  3
dm_crypt               23172  1
bnep                   19543  2
rfcomm                 69509  8
uvcvideo               81065  0
arc4                   12608  2
videobuf2_vmalloc      13216  1 uvcvideo
intel_rapl             18783  0
iwldvm                236430  0
x86_pkg_temp_thermal    14205  0
intel_powerclamp       18786  0
btusb                  32448  0
videobuf2_memops       13362  1 videobuf2_vmalloc
videobuf2_core         59104  1 uvcvideo
v4l2_common            15682  1 videobuf2_core
mac80211              660592  1 iwldvm
coretemp               13441  0
videodev              149725  3 uvcvideo,v4l2_common,videobuf2_core
media                  21963  2 uvcvideo,videodev
bluetooth             446190  22 bnep,btusb,rfcomm
kvm_intel             143592  0
kvm                   459835  1 kvm_intel
snd_hda_codec_hdmi     47547  1
crct10dif_pclmul       14307  0
6lowpan_iphc           18702  1 bluetooth
crc32_pclmul           13133  0
snd_hda_codec_conexant    23064  1
ghash_clmulni_intel    13230  0
snd_hda_codec_generic    68914  1 snd_hda_codec_conexant
aesni_intel           152552  10
snd_seq_midi           13564  0
snd_seq_midi_event     14899  1 snd_seq_midi
aes_x86_64             17131  1 aesni_intel
mei_me                 19742  0
lrw                    13287  1 aesni_intel
iwlwifi               183038  1 iwldvm
```

These are the kernel modules that are loaded. Use `modinfo` to get more info about a module; If you want.

If you need to add a module to your kernel (say a new driver for hardware) or remove it (uninstall a driver) you can use `rmmod` and `modprobe`.

Copy

```
# rmmod iwlwifi
```

And this is for installing the modules:

Copy

```
# insmod kernel/drivers/net/wireless/iwlwifi.ko
```

But nobody uses `insmod` because it does not understand dependencies and you need to give it the whole path to the module file. Instead, use the `modprobe` command:

Copy

```
# modprobe iwlwifi
```

> you can use `-f` switch to FORCE `rmmod` to remove the module even if it is in use

If you need to load some modules every time your system boots do one of the following:

1. Add their name to this file `/etc/modules`
2. Add their config files to the `/etc/modprobe.d/`

101.2 Boot the System

Weight: 3

Candidates should be able to guide the system through the booting process.

Key Knowledge Areas

- Provide common commands to the boot loader and options to the kernel at boot time
- Demonstrate knowledge of the boot sequence from BIOS/UEFI to boot completion
- Understanding of SysVinit and systemd
- Awareness of Upstart
- Check boot events in the log files

Terms

- dmesg
- journalctl
- BIOS
- UEFI
- bootloader
- kernel
- initramfs
- init
- SysVinit
- systemd

The Boot Process

It is important to understand because at this stage, you have very little control over the system and you can not issue commands to troubleshoot much. You should have a good understanding of what is happening.

1. Motherboard Firmware does a PowerOnSelfTest(POST)
2. Motherboard loads the bootloader
3. Bootloader loads the Linux Kernel-based on its configs/commands
4. The Kernel loads and prepares the system (root filesystem) and runs the initialization program
5. Init program start the service such as web server, graphical interface, networking, etc.

As we discussed in the previous section (101.1), the Firmware on the motherboard can be BIOS or UEFI.

BIOS

Basic Input Output System

- Older
- Limited to one sector of the disk and needs a multi-stage bootloader

- Can start the bootloader from internal/external HDD, CD/DVD, USB Flash drive, Network server
- If booting from the HDD, the Master Boot Record will be used (1 sector)

UEFI

Unified Extensible Firmware Interface.

- Modern and fancy
- Specifies a special disk partition for the bootloader. Called EFI System Partition (ESP)
- ESP is FAT and mounted on `/boot/efi` and bootloader files has `.efi` extensions

  You can check `/sys/firmware/efi` to see if you are using a UEFI system or not

Bootloader

Bootloader initializes the minimum hardware needed to boot the system. then, it locates and runs the OS.

Technically you can point your UEFI to run anything you want but typically under GNU/Linux systems, we use GRUB. GRUB can be used to run any specific program you need but it generally runs the OS.

Kernel

The Kernel is the core of your operating system. it basically is LINUX itself. Your bootloader loads the kernel in the memory and runs it. But the kernel needs some initial info to start; Things like drivers which are necessary to work with the hardware. Those are stored in `initrd` or `initramfs` alongside the kernel and used during the boot.

You can also send parameters to the kernel during the boot using the Grub configs. For example, sending a 1 or S will result the system booting in single-user mode (recovery). Or you can force your graphics to work in 1024×768x24 mode by passing `vga=792` to the Kernel during the boot.

dmesg

Linux will show you the boot process logs during the boot. Some desktop systems hide this behind a fancy splash screen which you can hide using the `Esc` key or press `Ctrl+Alt+F1`.

due to several reasons which are beyond the scope of this course, the kernel saves it's own logs into the "Kernel Ring Buffer". after the compilation of the boot process, the syslog daemon collects the *boot logs* and stores them in `/var/log/dmesg`.

to view all the logs including what has been logged after the boot process we use the `dmesg` command.

We can also use `journalctl -k` to check Kernel logs or use `journalctl -b` to check for boot logs (or even use `journalctl -u kernel` to see all previous logs too).

In addition to these, most systems keep the boot logs in a text-like file too and they can be found in `/var/log/boot` or `/var/log/boot.log` in Debian or Red-Hat based systems, respectively.

/var/log/messages

After the init process comes up, syslog daemon will log messages. It has timestamps and will persist during restarts.

- The Kernel is still logging its messages in Kernel Buffer Ring
- in some systems, it might be called `/var/log/syslog`

- there are many other logs at `/var/log`

init

When the Kernel initialization is finished, its time to start other programs. To do so, the Kernel runs the Initialization Daemon process, and it takes care of starting other daemons, services, subsystems and programs. Using the init system one can say "I need service A and then service B. Then I need C and D and E but do not start D unless the A and B are running". The system admin can use the init system to stop and start the services later.

There are different init systems:

- **SysVinit** is based on Unix System V. Not being used much anymore but people loved it because it followed Unix philosophies. you may see it on older machines or even on recently installed ones.
- **upstart** was an event-based replacement for the traditional init daemon developed by Canonical (The people behind Ubuntu). The goal of the project was to build a replacement for SysV when it got released in 2007. eventually, the project got discontinued due to the wide adoption of Systemd. Even Ubuntu uses Systemd these days, but upstart still can be found in google's ChromeOS.
- **Systemd** is the new replacement. It is hated by Linux elitists for not following Unix principles but it's widely adopted by major distros. It can start services in parallel and do lots of fancy stuff!

The init process had the ID of 1 and you can find it by running the

Copy

```
# which init
/sbin/init
# readlink -f /sbin/init
/usr/lib/systemd/systemd
# ps -p 1
PID TTY TIME     CMD
1   ?   00:00:06 systemd
```

You can check the hierarchy of processes using the `pstree` command.

Copy

```
pstree
```

systemd

Is new, loved, and hated. Lots of new ideas but not following some of the beloved UNIX principles (say.. not saving logs in a text file or trying to help you too much but asking for the root password when you are not running commands with sudo). It lets us run services if the hardware is connected, in time intervals, if another service is started, and ...

The systemd is made around **units**. A unit can be a service, group of services, or an action. Units do have a name, a type, and a configuration file. There are 12 unit types: automount, device, mount, path, scope, service, slice, snapshot, socket, swap, target & timer.

We use `systemctl` to work with these units and `journalctl` to see the logs.

Copy

```
# systemctl list-units
# systemctl list-units --type=target
# systemctl get-default # default target (groups of services are started via target unit files)
```

The units can be found in these places (sorted by priority):

1. `/etc/systemd/system/`
2. `/run/systemd/system/`
3. `/usr/lib/systemd/system`

Copy

```
# systemctl list-unit-files
# systemctl cat ntpd.service
# systemctl cat graphical.target
```

We can use these commands to work with services:

Copy

```
# systemctl stop sshd
# systemctl start sshd
# systemctl status sshd
# systemctl is-active sshd
# systemctl is-failed sshd
```

```
# systemctl restart sshd
# systemctl reload sshd # re-reads the configuration of the service configs
# systemctl daemon-reload sshd # re-reads the configuration of the systemd configs of this service
# systemctl enable sshd
# systemctl disable sshd
```

there are other commands too:

Copy

```
# systemctl is-system-running # running, degraded, maintenance, initializing, starting, stopping
# systemctl --failed
```

to check the logs, we have to use the `journalctl` utility:

Copy

```
# journalctl # show all journal
# journalctl --no-pager # do not use less
# journalctl -n 10 # only 10 lines
# journalctl -S -1d # last 1 day
# journalctl -xe # last few logs
# journalctl -u ntp # only ntp unit
# journalctl _PID=1234
```

SysV

Is the older init system. Still can be used on many systems. The control files are located at `/etc/init.d/` and are closer to the general bash scripts. In many cases you can call like:

Copy

```
/etc/init.d/ntpd status
/etc/init.d/ntpd stop
/etc/init.d/ntpd start
/etc/init.d/ntpd restart
```

We will speak more about runlevels on 101.3.

101.3 Change runlevels / boot targets and shutdown or reboot the system

Weight: 3

Description: Candidates should be able to manage the SysVinit runlevel or systemd boot target of the system. This objective includes changing to single-user mode, and shutdown or rebooting the system. Candidates should be able to alert users before switching runlevels / boot targets and properly terminate processes. This objective also includes setting the default SysVinit runlevel or systemd boot target. It also includes awareness of Upstart as an alternative to SysVinit or systemd.

Key Knowledge Areas:

- Set the default runlevel or boot target.
- Change between runlevels / boot targets including single-user mode.
- Shutdown and reboot from the command line.
- Alert users before switching runlevels / boot targets or other major system events.
- Properly terminate processes.
- Awareness of acpi.

The following is a partial list of the used files, terms and utilities:

- `/etc/inittab`
- shutdown
- init
- `/etc/init.d/`
- telinit
- systemd

- systemctl
- `/etc/systemd/`
- `/usr/lib/systemd/`
- wall

runlevels

Runlevels define what tasks can be accomplished in the current state (or runlevel) of a Linux system. Think of it as different stages of *being alive*.

systemd

On systemd, we have different targets which are groups of services:

Copy

```
root@debian:~# systemctl list-units --type=target # On a Debian machine
  UNIT                LOAD   ACTIVE SUB    DESCRIPTION
------------------------------------------------------------
  basic.target        loaded active active Basic System
  cryptsetup.target   loaded active active Local Encrypted Volumes
  getty.target        loaded active active Login Prompts
  graphical.target    loaded active active Graphical Interface
  local-fs-pre.target loaded active active Local File Systems (Pre)
  local-fs.target     loaded active active Local File Systems
  multi-user.target   loaded active active Multi-User System
  network.target      loaded active active Network
  paths.target        loaded active active Paths
  remote-fs.target    loaded active active Remote File Systems
  slices.target       loaded active active Slices
  sockets.target      loaded active active Sockets
  sound.target        loaded active active Sound Card
  swap.target         loaded active active Swap
  sysinit.target      loaded active active System Initialization
  time-set.target     loaded active active System Time Set
  time-sync.target    loaded active active System Time Synchronized
  timers.target       loaded active active Timers
```

And we can check the default one or get the status of each of them:

Copy

```
root@debian:~# systemctl get-default
graphical.target

root@debian:~# systemctl status multi-user.target
● multi-user.target - Multi-User System
     Loaded: loaded (/lib/systemd/system/multi-user.target; static)
     Active: active since Sat 2022-05-07 11:58:36 EDT; 4h 24min left
       Docs: man:systemd.special(7)
```

It is also possible to *isolate* any of the targets or move to two special targets too:

1. `rescue`: Local file systems are mounted, no networking, and root-user only (*maintenance* mode)
2. `emergency`: Only the root file system and in read-only mode, No networking and root-user only(*maintenance* mode)
3. `reboot`
4. `halt`: Stops all processes and halts CPU activities

5. `poweroff`: Like halt but also sends an ACPI shutdown signal (No lights!)

Copy

```
# systemctl isolate emergency
Welcome to emergency mode! After logging in, type "journalctl -xb" to view system logs, "systemctl reboot" to reboot, "systemctl default" or '
Give root password for maintenance
(or type Control-D to continue):
#
# systemctl is-system-running
maintenance
```

SysV runlevels

On SysV we were able to define different stages. On a Red Hat-based system we usually had 7:

- 0- Shutdown
- 1- Single-user mode (recovery); Also called S or s
- 2- Multi-user without networking
- 3- Multi-user with networking
- 4- to be customized by the admin
- 5- Multi-user with networking and graphics
- 6- Reboot

And in Debian based system we had:

- 0- Shutdown
- 1- Single-user mode
- 2- Multi-user mode with graphics
- 6- Reboot

Checking status and setting defaults

You can check your current runlevel with `runlevel` command. It comes from SysV era but still works on systemd systems. The default was in `/etc/inittab`

Copy

```
grep "^id:" /etc/inittab #on initV systems
id:5:initdefault:
```

It can also be done on grub kernel parameters.

Or using the runlevel and `telinit` command.

Copy

```
# runlevel
N 3
# telinit 5
# runlevel
3 5
# init 0 # shutdown the system
```

You can find the files in `/etc/init.d` and runlevels in `/etc/rc[0-6].d` directories where S indicates Start and K indicates Kill.

On systemd, you can find the configs in:

- `/etc/systemd`
- `/usr/lib/systemd/`

As discussed in 101.2

/etc/inittab

Is being replaced by upstart and systemd but is still part of the exam.

Copy

```
#
# inittab       This file describes how the INIT process should be set up
#               the system in a certain run-level.
#
# Author:       Miquel van Smoorenburg, <miquels@drinkel.nl.mugnet.org>
#               Modified for RHS Linux by Marc Ewing and Donnie Barnes
```

```
#

# Default runlevel. The runlevels used by RHS are:
#   0 - halt (Do NOT set initdefault to this)
#   1 - Single-user mode
#   2 - Multiuser, without NFS (The same as 3, if you do not have networking)
#   3 - Full multiuser mode
#   4 - unused
#   5 - X11
#   6 - reboot (Do NOT set initdefault to this)
#
id:5:initdefault:

# System initialization.
si::sysinit:/etc/rc.d/rc.sysinit

l0:0:wait:/etc/rc.d/rc 0
l1:1:wait:/etc/rc.d/rc 1
l2:2:wait:/etc/rc.d/rc 2
l3:3:wait:/etc/rc.d/rc 3
l4:4:wait:/etc/rc.d/rc 4
l5:5:wait:/etc/rc.d/rc 5
l6:6:wait:/etc/rc.d/rc 6

# Trap CTRL-ALT-DELETE
ca::ctrlaltdel:/sbin/shutdown -t3 -r now

# When our UPS tells us power has failed, assume we have a few minutes
# of power left.  Schedule a shutdown for 2 minutes from now.
# This does, of course, assume you have powered installed and your
# UPS connected and working correctly.
pf::powerfail:/sbin/shutdown -f -h +2 "Power Failure; System Shutting Down"

# If power was restored before the shutdown kicked in, cancel it.
pr:12345:powerokwait:/sbin/shutdown -c "Power Restored; Shutdown Cancelled"


# Run gettys in standard runlevels
1:2345:respawn:/sbin/mingetty tty1
2:2345:respawn:/sbin/mingetty tty2
3:2345:respawn:/sbin/mingetty tty3
4:2345:respawn:/sbin/mingetty tty4
5:2345:respawn:/sbin/mingetty tty5
6:2345:respawn:/sbin/mingetty tty6

# Run xdm in runlevel 5
x:5:respawn:/etc/X11/prefdm -nodaemon
```

This is the format:

Copy

```
id:runlevels:action:process
```

- id: 2 or 3 chars
- runlevels: Which runlevel this commands refers to (empty means all)
- action: Respawn, wait, once, initdefault (default run level as seen above), ctrlaltdel (What to do with Ctrl+Alt+Delete)

All scripts are here:

Copy

```
ls -ltrh /etc/init.d
```

And start/stop on runlevels are controlled from these directories:

Copy

```
root@funlife:~# ls /etc/rc2.d/
```

Stopping the system

The preferred method to shut down or reboot the system is to use the `shutdown` command, which first sends a warning message to all logged-in users and blocks any further non-root logins. It then signals init to switch runlevels. The init process then sends all running processes a SIGTERM signal, giving them a chance to save data or otherwise properly terminate. After 1 minute or another delay, if specified, init sends a SIGKILL signal to forcibly end each remaining process.

- Default is a 1-minute delay and then going to runlevel 1
- `-h` will halt the system
- `-r` will reboot the system
- Time is `hh:mm` or n (minutes) or now
- Whatever you add, will be broadcasted to logged-in users using the `wall` command
- If the command is running, ctrl+c or the `shutdown -c` will cancel it

Copy

```
shutdown -r 60 Reloading updated kernel
```

for more advanced users:

- -t60 will delay 60 seconds between SIGTERM and SIGKILL
- if you cancel a shutdown, users will get the news

Halt, reboot, and poweroff

- The `halt` command halts the system.
- The `poweroff` command halts the system and then attempts to power it off.
- The `reboot` command halts the system and then reboots it.

  On most distros, these are symbolic links to the systemctl utility

Advanced Configuration and Power Interface (ACPI)

ACPI provides an open standard that operating systems can use to discover and configure computer hardware components, perform power management (e.g. putting unused hardware components to sleep), perform auto-configuration (e.g. Plug and Play, and hot-swapping), and perform status monitoring.

This subsystem lets OS commands (like shutdown) send signals to the computer which results in powering down of the whole PC. In older times we used to have these mechanical keyboards to do a *real* power down after the OS has done its shutdown and told us that "it is not safe to power down your computer".

Notifying users

It is good to be informed! Especially if the system is going down; Especially on a shared server. Linux has different tools for system admins to notify their users:

- `wall`: Sending *wall messages* to logged-in users
- `/etc/issue`: Text to be displayed on the tty terminal logins (before login)
- `/etc/issue.net`: Text to be displayed on the remote terminal logins (before login)
- `/etc/motd`: Message of the day (after login). Some companies add "Do not enter if you are not allowed" texts here for legal reasons.
- `mesg`: Command controls if you want to get wall messages or not. You can do `mesg n` and `who -T` will show mesg status. Note that `shutdown` wall messages do not respect the `mesg` status

  systemctl sends wall messages for emergency, halt, power-off, reboot, and rescue

102.1 Design hard disk layout

Weight: 2

Description: Candidates should be able to design a disk partitioning scheme for a Linux system.

Key Knowledge Areas

- Allocate filesystems and swap space to separate partitions or disks
- Tailor the design to the intended use of the system
- Ensure the `/boot` partition conforms to the hardware architecture requirements for booting
- Knowledge of basic features of LVM

The following is a partial list of the used files, terms, and utilities:

- `/` (root) filesystem
- `/var` filesystem
- `/home` filesystem
- `/boot` filesystem
- EFI System Partition (ESP)
- Swap space
- Mount points
- Partitions

Basics

Like any contemporary OS, Linux uses *files* and *directories* to operate. But unlike *Windows*, it does not use A:, C:, D:, etc. In Linux, everything is in **one big tree*, starting with `/` (called root). Any partition, disk, CD, USB, network drive, ... will be placed somewhere in this huge tree.

> Note: Most of external devices (USB, CD, ...) are mounted at `/media/` or `/mnt/` .

Unix directories

This might be your enlightening moment in your Linux journey. Understanding **Filesystem Hierarchy Standard (FHS)** can help you find your programs, configs, logs and more without having prior knowledge about them. The table below has been the standard the latest revision since 2015.

Directory	Description
bin	Essential command binaries
boot	Static files of the boot loader
dev	Device files
etc	Host-specific system configuration
home	Home directory of the users
lib	Essential shared libraries and kernel modules
media	Mount point for removable media
mnt	Mount point for mounting a filesystem temporarily
opt	Add-on application software packages
root	Home directory of the root user
sbin	Essential system binaries
srv	Data for services provided by this system
tmp	Temporary files, sometimes purged on each boot
usr	Secondary hierarchy
var	Variable data (logs, ...)

Partitions

In the Linux world, devices are defined at `/dev/` and for different types of disks, there are different naming conventions: - PATA (Obsolete): `/dev/hdc` - SATA (Serial ATA) & SCSI disks: `/dev/sda` - SD/emmc & bare NAND/NOR devices: `/dev/mmcblk0` & their partitions are available as: `/dev/mmcblk0p0` - NVME drives: `/dev/nvme0` & their partitions are available as: `/dev/nvme0n1`

You have to *PARTITION* the disks, that is to create smaller parts on a big disk. These are self-contained sections on the main drive. OS sees these as standalone disks. We recognize them as: - /dev/sd**a**1 (first partition of the first SCSI disk) - /dev/hd**b**3 (3rd partition on the second disk)

BIOS systems were using MBR and could have up to 4 partitions on each disk, although instead of creating 4 Primary partitions, you could create an Extended partition and define more Logical partitions inside it.

> Note: an Extended partition is just an empty box for creating Logical partitions inside it.

So:

- `/dev/sda3` is the 3rd primary partition on the first disk
- `/dev/sdb5` is the first logical partition on the second disk
- `/dev/sda7` is the 3rd logical partition of the first physical disk

UEFI systems use GUID Partition Table (GPT) which supports 128 partitions on each device.

> If you define an extended partition on a BIOS system, that will be `/dev/sdx5` (1-4 for primary, and the first extended will be 5).

Linux systems can **mount** these partitions on different paths. Say you can have a separated disk with one huge partition for your `/home` and another one for your `/var/logs/`.

Copy

```
# fdisk /dev/sda
Welcome to fdisk (util-linux 2.25.1).
Changes will remain in memory only, until you decide to write them.
Be careful before using the write command.


Command (m for help): p
Disk /dev/sda: 298.1 GiB, 320072933376 bytes, 625142448 sectors
Units: sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
Disklabel type: dos
Disk identifier: 0x000beca1

Device     Boot     Start       End   Sectors   Size Id Type
/dev/sda1  *         2048  43094015  43091968  20.6G 83 Linux
/dev/sda2        43094016  92078390  48984375  23.4G 83 Linux
/dev/sda3        92080126 625141759 533061634 254.2G  5 Extended
/dev/sda5        92080128 107702271  15622144   7.5G 82 Linux swap / Solaris
/dev/sda6       107704320 625141759 517437440 246.8G 83 Linux
```

The newer **GUID Partition Table (or GPT)** solves these problems. If you format your disk with GPT you can have 128 primary partitions (no need for extended and logical).

Commands

parted

Copy

```
jadi@funlife:~$ sudo parted /dev/sda p
Model: ATA ST320LT000-9VL14 (scsi)
Disk /dev/sda: 320GB
Sector size (logical/physical): 512B/512B
Partition Table: msdos
Disk Flags:

Number  Start   End     Size    Type      File system     Flags
 1      1049kB  22.1GB  22.1GB  primary   ext4            boot
 2      22.1GB  47.1GB  25.1GB  primary   ext4
 3      47.1GB  320GB   273GB   extended
 5      47.1GB  55.1GB  7999MB  logical   linux-swap(v1)
 6      55.1GB  320GB   265GB   logical
```

fdisk

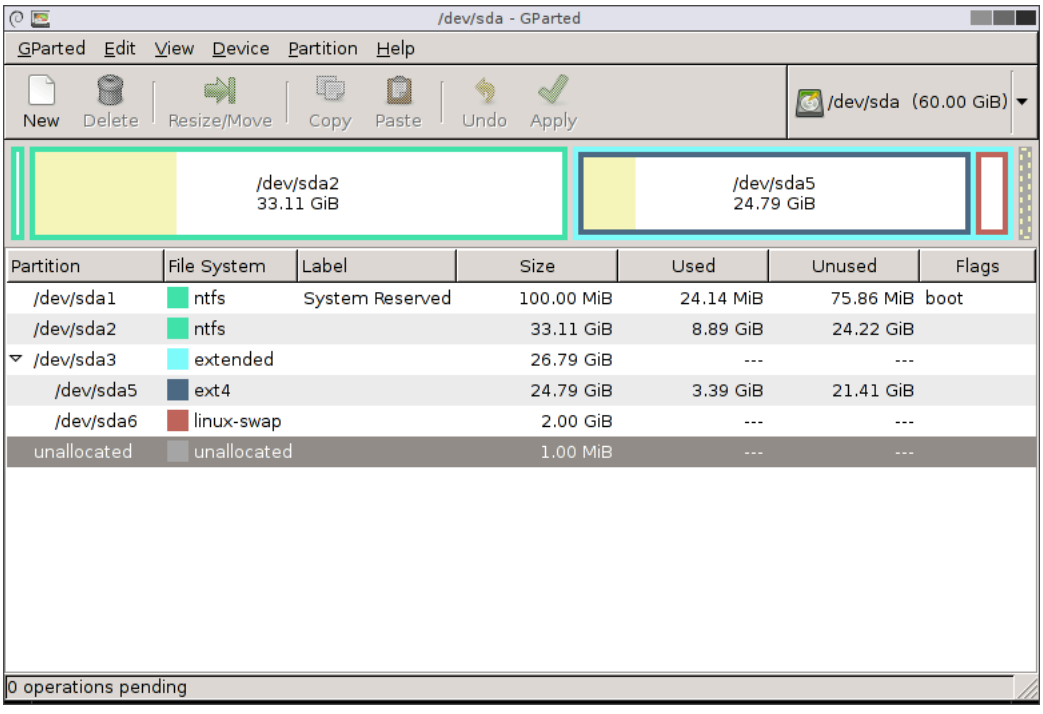
Copy

```
# sudo fdisk /dev/sda
[sudo] password for jadi:

Welcome to fdisk (util-linux 2.25.1).
Changes will remain in memory only, until you decide to write them.
Be careful before using the write command.


Command (m for help): p
Disk /dev/sda: 298.1 GiB, 320072933376 bytes, 625142448 sectors
Units: sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
Disklabel type: dos
Disk identifier: 0x000beca1

Device     Boot     Start       End   Sectors   Size Id Type
/dev/sda1  *         2048  43094015  43091968  20.6G 83 Linux
/dev/sda2        43094016  92078390  48984375  23.4G 83 Linux
/dev/sda3        92080126 625141759 533061634 254.2G  5 Extended
/dev/sda5        92080128 107702271  15622144   7.5G 82 Linux swap / Solaris
/dev/sda6       107704320 625141759 517437440 246.8G 83 Linux
```

Note: parted does not understands GPT

gparted

A graphical tool for managing disks and partitions.



LVM

In many cases, you need to resize your partitions or even install new disks and *add* them to your current mount points; Increasing the total size. LVM is designed for this.

LVM helps you create one partition from different disks and add or remove space from/to them. The main concepts are:

- Physical Volume (PV): A whole drive or a partition. It is better to define partitions and **not use whole disks - unpartitioned**.
- Volume Groups (VG): This is the collection of one or more **PV**s. OS will see the vg as one big disk. PVs in one VG, can have different sizes or even be on different physical disks.
- Logical Volumes (LV): OS will see lvs as partitions. You can format an LV with your OS and use it.

Design Hard disk layout

Disk layout and allocation partitions to directories depend on your usage. First, we will discuss *swap* and *boot* and then will see three different cases.

swap

Swap in Linux works like an extended memory. The Kernel will *page* memory to this partition/file. It is enough to format one partition with **swap file system** and define it in `/etc/fstab` (you will see this later in 104 modules).

> Note: There is no strict formula for swap size. People used to say "double the ram but not more than 8GB". On recent machines with SSDs, some say "RAM + 2" (Hibernation + some extra) or "RAM * 2" depending on your usage.

/boot

Older Linux systems were not able to handle HUGE disks during boot (say Terabytes) so the `/boot` partition was separated. this separation comes in handy in situations such as recovering broken systems. you can even make `/boot` read-only. Most of the time, having 100MB for `/boot` is enough. This partition can be on a different disk or a separated partition.

This partition should be accessible by BIOS/UEFI during boot (No network drive).

On UEFI systems, there is a `/boot/efi` mount point called EFI (Extensible Firmware Interface) system partition or ESP. This contains the bootloader and kernel and should be accessible by the UEFI firmware during boot.

Case one: Desktop computer

On a desktop computer, it is good to have one swap, one `/boot`, and allocate all other space to `/` (root).

Case two: Network workstation

As any other system `/boot` should be local (a physical disk is connected to the machine) and most of the time, the `/` (root file system) is also local. But in a network station, `/home` can be mounted from a network drive (NFS, SMB, SSH, ..). This lets users sit at any station, login, and have their own home mounted from a network drive. Swap can be mounted from network or local.

Case three: Server

On servers `/boot` is still local and based on usage, `/home` can be local or network. In many cases, we separate the `/var` because logs and many other files are there and being updated so it is good to separate it or even put it on more advanced storage (like RAID disks to prevent data loss). Some people also separate the `/usr` and write-protect it (read-only file systems) or even mount the `/usr` from the network so they can change/update one file on the network storage and all the servers will use the new file (You remember? `/usr` contains important executables like Apache webserver).

Bonus! know about zram

Here I'll review the 3 different methods to add a *swap* to your OS. We will have a look at 3 different distros:

- Debian 11: Uses a swap partition
- Ubuntu 22.04: Uses a swap file
- Fedora 36: uses `zram` In short, `zram` is a virtual disk on your RAM which can be used as swap space or be mounted anywhere you like; A common example is `\tmp`. Let's see.

102.2 Install a boot manager

Weight: 2

Description: Candidates should be able to select, install and configure a boot manager.

Objectives

- Providing alternative boot locations and backup boot options.
- Install and configure a boot loader such as GRUB Legacy.
- Perform basic configuration changes for GRUB 2.
- Interact with the boot loader

Terms and Utilities

- `menu.lst`, `grub.cfg` and `grub.conf`
- grub-install
- grub-mkconfig
- MBR

Boot overview

Most systems use BIOS or UEFI. When on BIOS, the system will do a self test called POST (Power-On Self-Test). Then it will hand over the boot process to the first sector of Master Boot Record (MBR) which is track (Cylinder) 0, side (Head) 0 and Sector 1 of the first disk.

MBR is only 512 bytes so we need a *smart bootloader* to handle larger boot managers and even multiple systems. Some of these boot loaders are LILO, GRUB and GRUB2.

If the system is using UEFI, the hardware will follow the UEFI stages. They start with a security phase and will continue till the end phase where the UEFI looks for an EFI System Partition, which is just a FAT32 partition (Usually the first one, but that's implementation-defined) with PE executables and runs them.

In both cases, the binary starts the boot loader. It might be a complete bootloader on `/boot/efi/` of your computer or a small loader for the main grub on the MBR or a windows loader or even a chainloader.

> Chain Loading is when a boot loader, loads another boot loader. This is done when a Linux bootloader needs to start a Windows system.

GRUB

GRUB (**GR**and **U**nified **B**ootloader) started to replace the older LILO. The first version (1) is called Grub Legacy and started in 1999. The 2nd version started in 2005 and is a complete rewrite of version 1.

It's a menu-based system where you can choose which Kernel or chainloader to boot. It is also possible to edit the menus on the fly or give direct commands from a command line.

Grub Legacy

Usually the GRUB v1 (actually 0.9) is installed in `/boot/grub`. Its main configuration is in `/boot/grub/menu.lst` but nowadays some distros (including RedHat Based ones) link this to the `/boot/grub/grub.conf`.

A sample `menu.lst` / `grub.conf` file for GRUB legacy consists of two sections. The first section contains global configs and the 2nd part defines different kernel/initram or chainloader options.

The global configs are:

Config	Description
#	Comment
color	Foreground and background colors for normal and active items
default	Which boot menu item is the default
fallback	Which boot menu should be used if the *default* fails
hiddenmenu	Hide the menu options
splashimage	Show this image in the background!
timeout	Wait this much and then start the default
password	Security is important! Will ask this password
savedefault	Remember the last booted item

On the second part of the config, we have these:

Config	Description
title	Defines the section name
root	Disk and partition where `/boot` directory is. In the form of (hddrive, partition), say (hd0, 0) or (hd0, msdos0)
kernel	Kernel image file name in `/boot`
initrd	Initramfs file in `/boot`
rootnoverify	Defines a non-Linux root partition
chainloader	Another file will act as stage 1 loader. Used for booting Windows systems

Here you can see a sample of GRUB-Legacy config:

Copy

```
# grub.conf generated by anaconda
#
# Note that you do not have to rerun grub after making changes to this file
# NOTICE:  You do not have a /boot partition.  This means that
#          all kernel and initrd paths are relative to /, eg.
#          root (hd0,5)
#          kernel /boot/vmlinuz-version ro root=/dev/sda6
#          initrd /boot/initrd-version.img
#boot=/dev/sda6
default=1
timeout=10
splashimage=(hd0,5)/boot/grub/splash.xpm.gz
#hiddenmenu
password --md5 $1$RW1VW/$4XGAklxB7/GJk0uO47Srx1
title Upgrade to Fedora 11 (Leonidas)
    kernel /boot/upgrade/vmlinuz preupgrade \
```

```
      repo=hd::/var/cache/yum/preupgrade stage2=\
      hd:UUID=8b4c62e7-2022-4288-8995-5eda92cd149b:/boot/upgrade/install.img \
      ks=hd:UUID=8b4c62e7-2022-4288-8995-5eda92cd149b:/boot/upgrade/ks.cfg
    initrd /boot/upgrade/initrd.img
title Fedora (2.6.26.8-57.fc8)
    root (hd0,5)
    kernel /boot/vmlinuz-2.6.26.8-57.fc8 ro root=LABEL=FEDORA8 rhgb quiet
    initrd /boot/initrd-2.6.26.8-57.fc8.img
title Fedora (2.6.26.6-49.fc8)
    root (hd0,5)
    kernel /boot/vmlinuz-2.6.26.6-49.fc8 ro root=LABEL=FEDORA8 rhgb quiet
    initrd /boot/initrd-2.6.26.6-49.fc8.img
title GRUB Menu
    rootnoverify (hd0,1)
    chainloader +1
title Windows
    rootnoverify (hd0,0)
    chainloader +1
```

GRUB (legacy) commands

After creating the configuration, you need to install the grub on a disk MBR. To do this you can use two different formats:

Copy

```
# grub-install /dev/fd0
# grub-install '(fd0)'
```

Just like any other boot manager, you can install grub on a CD, floppy, MBR (`/dev/sda`, `/dev/sdb`, ..) or a partition (`/dev/sdb2`, `/dev/sda6`, ..). But if you want to install it on anywhere other than the MBR, use a chainloader to point your boot sequence toward it.

If you needed to change or reconfigure anything during the startup, just press the `e` on that item and you'll get an interactive editing environment. Press **Enter** when done and `b` for boot.

Interacting with GRUB Legacy

If you press `c` on the grub menu, you will go into the *GRUB Command Line* or *GRUB shell*. There you can type commands like `root` and `kernel` and `initrd` and boot the system with `boot` or press the `Esc` key to return back to the menu.

GRUB2

This is the most common boot loader these days. On BIOS systems it is installed on `/boot/grub/` or `/boot/grub2/` and under UEFI it goes in `/boot/efi/EFI/distro-name/` (say `/boot/efi/EFI/fedora/`). GRUB2's configuration file is called `grub.cfg`.

Here is a simplified `grub.cfg`:

Copy

```
set default="0"
menuentry "Fedora" {
  set root=(hd0,1)
  linux /boot/vmlinuz-5.10.0-9-arm64 ro quiet
  initrd /boot/initrd.img-5.10.0-9-arm64
}
menuentry "Windows" {
```

```
  chainloader (hd1,msdos2)+1
}
```

As you can see, GRUB uses Linux-style numbering for partitions, so the first partition on the first hard disk is `(hd0,1)` or `(hd0,msdos1)` for DOS partitions or `(hd0,gpt1)` for GPT drives.

Here you can see some of the options:

Option	Description
menuentry	Defines a new menuentry
set root	Defines the root where `/boot` located
linux, linux16	Defines the location of the Linux kernel on BIOS systems
linuxefi	Defines the Linux kernel on UEFI systems
initrd	Defines the initramfs image for BIOS systems
initrdefi	Defines the initramfs image for UEFI systems

And here is a real-world `grub.cfg`:

Copy

```
#
# DO NOT EDIT THIS FILE
#
# It is automatically generated by grub-mkconfig using templates
# from /etc/grub.d and settings from /etc/default/grub
#

### BEGIN /etc/grub.d/00_header ###
if [ -s $prefix/grubenv ]; then
  set have_grubenv=true
  load_env
fi
if [ "${next_entry}" ] ; then
   set default="${next_entry}"
   set next_entry=
   save_env next_entry
   set boot_once=true
else
   set default="0"
fi

if [ x"${feature_menuentry_id}" = xy ]; then
  menuentry_id_option="--id"
else
  menuentry_id_option=""
fi

export menuentry_id_option

if [ "${prev_saved_entry}" ]; then
  set saved_entry="${prev_saved_entry}"
  save_env saved_entry
  set prev_saved_entry=
  save_env prev_saved_entry
  set boot_once=true
fi

function savedefault {
  if [ -z "${boot_once}" ]; then
    saved_entry="${chosen}"
    save_env saved_entry
  fi
}
function load_video {
  if [ x$feature_all_video_module = xy ]; then
    insmod all_video
  else
    insmod efi_gop
    insmod efi_uga
    insmod ieee1275_fb
    insmod vbe
    insmod vga
    insmod video_bochs
    insmod video_cirrus
  fi
}

if [ x$feature_default_font_path = xy ] ; then
```

```
    font=unicode
  else
  insmod part_gpt
  insmod ext2
  if [ x$feature_platform_search_hint = xy ]; then
    search --no-floppy --fs-uuid --set=root  df2d9e14-f1e9-4b1b-95d4-0167caa2461e
  else
    search --no-floppy --fs-uuid --set=root df2d9e14-f1e9-4b1b-95d4-0167caa2461e
  fi
      font="/usr/share/grub/unicode.pf2"
  fi

  if loadfont $font ; then
    set gfxmode=auto
    load_video
    insmod gfxterm
    set locale_dir=$prefix/locale
    set lang=en_US
    insmod gettext
  fi
  terminal_output gfxterm
  if [ "${recordfail}" = 1 ] ; then
    set timeout=30
  else
    if [ x$feature_timeout_style = xy ] ; then
      set timeout_style=menu
      set timeout=5
    # Fallback normal timeout code in case the timeout_style feature is
    # unavailable.
    else
      set timeout=5
    fi
  fi
  ### END /etc/grub.d/00_header ###

  ### BEGIN /etc/grub.d/05_debian_theme ###
  insmod part_gpt
  insmod ext2
  if [ x$feature_platform_search_hint = xy ]; then
    search --no-floppy --fs-uuid --set=root  df2d9e14-f1e9-4b1b-95d4-0167caa2461e
  else
    search --no-floppy --fs-uuid --set=root df2d9e14-f1e9-4b1b-95d4-0167caa2461e
  fi
  insmod png
  if background_image /usr/share/desktop-base/homeworld-theme/grub/grub-4x3.png; then
    set color_normal=white/black
    set color_highlight=black/white
  else
    set menu_color_normal=cyan/blue
    set menu_color_highlight=white/blue
  fi
  ### END /etc/grub.d/05_debian_theme ###

  ### BEGIN /etc/grub.d/10_linux ###
  function gfxmode {
      set gfxpayload="${1}"
  }
  set linux_gfx_mode=
  export linux_gfx_mode
  menuentry 'Debian GNU/Linux' --class debian --class gnu-linux --class gnu --class os $menuentry_id_option 'gnulinux-simple-df2d9e14-f1e9-4b1b
      load_video
      insmod gzio
      if [ x$grub_platform = xxen ]; then insmod xzio; insmod lzopio; fi
      insmod part_gpt
      insmod ext2
      if [ x$feature_platform_search_hint = xy ]; then
        search --no-floppy --fs-uuid --set=root  df2d9e14-f1e9-4b1b-95d4-0167caa2461e
      else
        search --no-floppy --fs-uuid --set=root df2d9e14-f1e9-4b1b-95d4-0167caa2461e
      fi
      echo    'Loading Linux 5.10.0-9-arm64 ...'
      linux   /boot/vmlinuz-5.10.0-9-arm64 root=UUID=df2d9e14-f1e9-4b1b-95d4-0167caa2461e ro  quiet
      echo    'Loading initial ramdisk ...'
      initrd  /boot/initrd.img-5.10.0-9-arm64
  }
  submenu 'Advanced options for Debian GNU/Linux' $menuentry_id_option 'gnulinux-advanced-df2d9e14-f1e9-4b1b-95d4-0167caa2461e' {
      menuentry 'Debian GNU/Linux, with Linux 5.10.0-9-arm64' --class debian --class gnu-linux --class gnu --class os $menuentry_id_option 'gnu
          load_video
          insmod gzio
          if [ x$grub_platform = xxen ]; then insmod xzio; insmod lzopio; fi
          insmod part_gpt
```

```
        insmod ext2
        if [ x$feature_platform_search_hint = xy ]; then
          search --no-floppy --fs-uuid --set=root  df2d9e14-f1e9-4b1b-95d4-0167caa2461e
        else
          search --no-floppy --fs-uuid --set=root df2d9e14-f1e9-4b1b-95d4-0167caa2461e
        fi
        echo    'Loading Linux 5.10.0-9-arm64 ...'
        linux   /boot/vmlinuz-5.10.0-9-arm64 root=(hd0,1) ro  quiet
        echo    'Loading initial ramdisk ...'
        initrd  /boot/initrd.img-5.10.0-9-arm64
    }
    menuentry 'Debian GNU/Linux, with Linux 5.10.0-8-arm64 (recovery mode)' --class debian --class gnu-linux --class gnu --class os $menuentry
        load_video
        insmod gzio
        if [ x$grub_platform = xxen ]; then insmod xzio; insmod lzopio; fi
        insmod part_gpt
        insmod ext2
        if [ x$feature_platform_search_hint = xy ]; then
          search --no-floppy --fs-uuid --set=root  df2d9e14-f1e9-4b1b-95d4-0167caa2461e
        else
          search --no-floppy --fs-uuid --set=root df2d9e14-f1e9-4b1b-95d4-0167caa2461e
        fi
        echo    'Loading Linux 5.10.0-8-arm64 ...'
        linux   /boot/vmlinuz-5.10.0-8-arm64 root=UUID=df2d9e14-f1e9-4b1b-95d4-0167caa2461e ro single
        echo    'Loading initial ramdisk ...'
        initrd  /boot/initrd.img-5.10.0-8-arm64
    }
}

### END /etc/grub.d/10_linux ###

### BEGIN /etc/grub.d/20_linux_xen ###

### END /etc/grub.d/20_linux_xen ###

### BEGIN /etc/grub.d/30_os-prober ###
### END /etc/grub.d/30_os-prober ###

### BEGIN /etc/grub.d/30_uefi-firmware ###
menuentry 'System setup' $menuentry_id_option 'uefi-firmware' {
    fwsetup
}
### END /etc/grub.d/30_uefi-firmware ###

### BEGIN /etc/grub.d/40_custom ###
# This file provides an easy way to add custom menu entries.  Simply type the
# menu entries you want to add after this comment.  Be careful not to change
# the 'exec tail' line above.
### END /etc/grub.d/40_custom ###

### BEGIN /etc/grub.d/41_custom ###
if [ -f  ${config_directory}/custom.cfg ]; then
  source ${config_directory}/custom.cfg
elif [ -z "${config_directory}" -a -f  $prefix/custom.cfg ]; then
  source $prefix/custom.cfg;
fi
### END /etc/grub.d/41_custom ###
```

GRUB2 commands

The installation is done with `grub-install /dev/sda` and after changing the config files, you need to issue `grub2-mkconfig` or `grub-mkconfig`. It reads the configuration files from `/etc/grub.d/` and `/etc/default/grub/` and create the `grub.cfg` file based on them. You run it like this:

Copy

```
grub2-mkconfig > /boot/grub2/grub.cfg
```

or

Copy

```
grub2-mkconfig -o /boot/grub2/grub.cfg
```

There is also a command called `update-grub` as a frontend to `grub-mkconfig` which runs `grub-mkconfig -o /boot/grub/grub.cfg`

> Please note that on some modern distros, you have both `grub` and `grub2` commands available for compatibility reasons, and one links to the other.

Interacting with GRUB2

If you press `c` on the grub menu, you will go into the *GRUB Command Line* or *GRUB shell*. There you can type commands like `root` and `kernel` and `initrd` and boot the system with `boot` or press the `Esc` key to return back to the menu.

Kernel boot parameters

In the above configs, we sent some parameters to the kernel like this:

Copy

```
        linux   /boot/vmlinuz-5.10.0-9-arm64 root=/dev/sda1 ro  quiet
```

This tells the kernel to boot in *ReadOnly* mode and does not show lots of logs during the boot (*quiet*).

These are some of the other options you may use:

Option	Description
console=	Set the console
debug	Start in debug mode
init=	Run an specific program instead of the default init
initrd=	Use this initrd
ro	Mount the root filesystem read only
rw	Mount the root filesystem for read and write
root=	Use this as the root filesystem
selinux	Disable `selinux` on boot
single,S,1,Single	Boot in single user mode for troubleshooting (SysV)
systemd.unit=	Boot in this systemd target

102.3 Manage shared libraries

Weight: 1

Objectives

Candidates should be able to determine the shared libraries that executable programs depend on and install them when necessary.

- Identify shared libraries.
- Identify the typical locations of system libraries.
- Load shared libraries.

Terms and Utilities

- ldd
- ldconfig
- `/etc/ld.so.conf`
- LD_LIBRARY_PATH

Linking

When we write a program, we use libraries. For example, if you need to read text from standard input, you need to *link* a library that provides this. Think linking has two forms:

- **Static** Linking is when you add this library to your executable program. In this method, your program size is big because it has all the needed libraries. One good advantage is your program can be run without being dependent on other programs/libraries.
- **Dynamic** Linking is when you just say in your program "We need this and that library to run this program". This way your program is smaller but you need to install those libraries separately. This makes programs more secure (Because libraries can be updated centrally, and more advanced any improvement in a library will improve the whole program), and smaller.

Linux dynamic libraries have names like `libLIBNAME.so.VERSION` and are located at places like `/lib*/` and `/usr/lib*/`. On Windows, we call them Dynamic Linked Libraries (DLLs).

> Dynamic linking is also called **shared** libraries because all the programs are sharing one library which is separately installed.

What libraries do I need

Libraries related to system utilities are installed in `/lib` and `/lib64` (for 32bit and 64bit libraries) and libraries installed by other software will be located at `/usr/lib` and `/usr/lib64`.

ldd

The `ldd` command helps you find:

- If a program is dynamically or statically linked
- What libraries a program needs

Let's have a look at two files:

Copy

```
[jadi@fedora ~]$ ldd /sbin/ldconfig
    not a dynamic executable

[jadi@fedora ~]$ ldd /bin/ls
    linux-vdso.so.1 (0x00007ffdd53eb000)
    libselinux.so.1 => /lib64/libselinux.so.1 (0x00007f5cbc7b0000)
    libcap.so.2 => /lib64/libcap.so.2 (0x00007f5cbc7a6000)
    libc.so.6 => /lib64/libc.so.6 (0x00007f5cbc5a5000)
    libpcre2-8.so.0 => /lib64/libpcre2-8.so.0 (0x00007f5cbc50f000)
    /lib64/ld-linux-x86-64.so.2 (0x00007f5cbc813es
```

As you can see, `ldd` tells us that the `/sbin/ldconfig` is not dynamically linked but shows us the libraries needed by `/bin/ls`.

Symbolic links for libraries

If you are writing a program and you use udev functions, you will ask for a library called *libudev.so.1*. But a Linux distro, might call its version of udev library *libudev.so.1.4.0*. How can we solve this problem? The answer is **symbolic links**; You will learn more about them in the next chapters but for short, a symbolic name is a new name for the same file.

I will check the same thing on my system. First I'll find where the `libudev.so.1` on my system is:

Copy

```
# locate libudev.so.1
/lib/i386-linux-gnu/libudev.so.1
/usr/lib64/libudev.so.1.7.3
```

And then will check that file:

Copy

```
# ls -la /lib/i386-linux-gnu/libudev.so.1
lrwxrwxrwx 1 root root    16 Nov 13 23:05 /lib/i386-linux-gnu/libudev.so.1 -> libudev.so.1.4.0
```

As you can see, this is a symbolic link pointing to the version of libudev I have installed (1.4.0) so even if a software says it needs libudev.so.1, my system will use its libusdev.so.1.4.0.

Dynamic library configs and cache

As with most other Linux tools, dynamic linking is also configured using a text config file. It is located at */etc/ld.so.conf*. On an Ubuntu system it just points to other config files in `/etc/ld.so.conf.d/` but all those lines can be included in the main file too:

Copy

```
[jadi@fedora ~]$ cat /etc/ld.so.conf
include ld.so.conf.d/*.conf
[jadi@fedora ~]$ ls /etc/ld.so.conf.d/
llvm13-x86_64.conf  pipewire-jack-x86_64.conf
[jadi@fedora ~]$ cat /etc/ld.so.conf.d/llvm13-x86_64.conf
/usr/lib64/llvm13/lib
```

The `ldconfig` commands processed all these files to make the loading of libraries faster. This command creates `ld.so.cache` to locate files that are to be dynamically loaded and linked.

> If you change the `ld.so.conf` (or sub-directories) you need to run `ldconfig`. Try it with `-v` switch to see the progress / data.

To close this section, let's run ldconfig with the **-p** switch to see what is saved in ld.so.cache:

Copy

```
[jadi@fedora ~]$ ldconfig -p | head
1373 libs found in cache `/etc/ld.so.cache'
    libzstd.so.1 (libc6,x86-64) => /lib64/libzstd.so.1
    libzmf-0.0.so.0 (libc6,x86-64) => /lib64/libzmf-0.0.so.0
    libzip.so.5 (libc6,x86-64) => /lib64/libzip.so.5
    libzhuyin.so.13 (libc6,x86-64) => /lib64/libzhuyin.so.13
    libzck.so.1 (libc6,x86-64) => /lib64/libzck.so.1
    libz.so.1 (libc6,x86-64) => /lib64/libz.so.1
    libyui.so.15 (libc6,x86-64) => /lib64/libyui.so.15
    libyui-mga.so.15 (libc6,x86-64) => /lib64/libyui-mga.so.15
    libyelp.so.0 (libc6,x86-64) => /lib64/libyelp.so.0
```

As you can see, this file tells the kernel that if anyone asks for `libzstd.so.1`, the `/lib64/libzstd.so.1` file should be loaded and used.

Where OS finds dynamic libraries

When a program needs a shared library, the system will search files in this order:

1. LD_LIBRARY_PATH environment variable
2. Programs PATH
3. `/etc/ld.so.conf` (Which might load more files from `/etc/ld.so.conf.d/` in its beginning or its end)
4. `/lib/`, `/lib64/`, `/usr/lib/`, `/usr/lib64/`

In some cases, you might need to override the default system libraries. Some examples are:

- You are running an old software which needs an old version of a library.
- You are developing a shared library and want to test it without installing it
- You are running a specific program (say from opt) which needs to access its own libraries

In these cases, you can point the environment variable **LD_LIBRARY_PATH** to the library you need to use and then run your program. A colon (:) separated list of directories will tell your program where to search for needed libraries **before** checking the libraries in `/etc/ld.so.cache`.

For example, if you give this command:

Copy

```
export  LD_LIBRARY_PATH=/usr/lib/myoldlibs:/home/jadi/lpic/libs/
```

And then run any command, the system will search `/usr/lib/myoldlibs` and then `/home/jadi/lpic/libs/` before going to the main system libraries (defined in `/etc/ld.so.cache`). . .

Loading dynamically

In the last part of this section, let's see how we can manually tell Linux to run a program using its *dynamic linker*. Its also called a dynamic loader and is used to load dynamic libraries needed by an executable. It might be called `ld` or `ld-linux`. You can find yours by running:

Copy

```
[jadi@fedora ~]$ locate ld-linux
/usr/lib64/ld-linux-x86-64.so.2
/usr/share/man/man8/ld-linux.8.gz
/usr/share/man/man8/ld-linux.so.8.gz
```

and run programs using it like this:

Copy

```
[jadi@fedora ~]$ /usr/lib64/ld-linux-x86-64.so.2 /usr/bin/ls
Desktop  Documents  Downloads  Music  Pictures  Public  Templates  tmp  Videos
```

Do you want to go deeper than LPIC1 and ask why you do not need to run `ld-linux` when running a command? Because the executable you are running is a Linux ELF executable and if you check its inside, you can see that it says "run this using ld-linux":

Copy

```
[jadi@fedora ~]$ readelf -Wl /usr/bin/ls

Elf file type is DYN (Position-Independent Executable file)
Entry point 0x6c00
There are 13 program headers, starting at offset 64

Program Headers:
  Type           Offset   VirtAddr           PhysAddr           FileSiz  MemSiz   Flg Align
  PHDR           0x000040 0x0000000000000040 0x0000000000000040 0x0002d8 0x0002d8 R   0x8
  INTERP         0x000318 0x0000000000000318 0x0000000000000318 0x00001c 0x00001c R   0x1
      [Requesting program interpreter: /lib64/ld-linux-x86-64.so.2]
  LOAD           0x000000 0x0000000000000000 0x0000000000000000 0x003798 0x003798 R   0x1000
  LOAD           0x004000 0x0000000000004000 0x0000000000004000 0x013c21 0x013c21 R E 0x1000
  LOAD           0x018000 0x0000000000018000 0x0000000000018000 0x0074d8 0x0074d8 R   0x1000
  [...]
```

💀 There is a *hack* here: *You can run any Linux executable even if its executable bit is not set!* just run it using `ld-linux` as we did a few lines above!

102.4 Use Debian package management

Weight: 3

Candidates should be able to perform package management using the Debian package tools.

- Install, upgrade and uninstall Debian binary packages.
- Find packages containing specific files or libraries which may or may not be installed.
- Obtain package information like version, content, dependencies, package integrity, and installation status (Whether or not the package is installed).
- Awareness of apt.

Terms and Utilities

- `/etc/apt/sources.list`
- dpkg
- dpkg-reconfigure
- apt-get
- apt-cache

Concept of the package management system

Some people think that on GNU/Linux we have to compile all the software we need manually. This is not the case in 99% of cases and never has been the case in the last 20 years. GNU/Linux is the predecessor of what we call the App Store these days. All major distros do have huge archives of pre-compiled software called their *repositories* and some kind of a **package manager** software that takes care of searching these repositories, installing software from them, finding dependencies, installing them, resolving conflicts, and updating the system and installed software. Debian-based distros use `.deb` files as their "packages" and use tools like `apt-get`, `dpkg`, `apt`, and other tools to manage them.

Debian packages are names like `NAME-VERSION-RELEASE_ARCHITECTURE.deb`; Say `tmux_3.2a-4build1_amd64.deb`.

Repositories

But where did this package come from? How does the OS know where to look for this deb package? The answer is **Repositories**. Each distro has its repository of packages. It can be on a Disk, A network drive, a collection of DVDs, or most commonly, a network address on the Internet.

On debian systems, the main configuration locations are: * `/etc/apt/sources.list` * `/etc/apt/sources.list.d/`

Copy

```
jadi@lpicjadi:~$ cat /etc/apt/sources.list
#deb cdrom:[Ubuntu 22.04 LTS _Jammy Jellyfish_ - Beta amd64 (20220329.1)]/ jammy main restricted

# See http://help.ubuntu.com/community/UpgradeNotes for how to upgrade to
# newer versions of the distribution.
deb http://us.archive.ubuntu.com/ubuntu/ jammy main restricted
# deb-src http://us.archive.ubuntu.com/ubuntu/ jammy main restricted

## Major bug fix updates produced after the final release of the
## distribution.
deb http://us.archive.ubuntu.com/ubuntu/ jammy-updates main restricted
# deb-src http://us.archive.ubuntu.com/ubuntu/ jammy-updates main restricted

## N.B. software from this repository is ENTIRELY UNSUPPORTED by the Ubuntu
## team. Also, please note that software in the universe WILL NOT receive any
## review or updates from the Ubuntu security team.
deb http://us.archive.ubuntu.com/ubuntu/ jammy universe
# deb-src http://us.archive.ubuntu.com/ubuntu/ jammy universe
deb http://us.archive.ubuntu.com/ubuntu/ jammy-updates universe
# deb-src http://us.archive.ubuntu.com/ubuntu/ jammy-updates universe

## N.B. software from this repository is ENTIRELY UNSUPPORTED by the Ubuntu
## team, and may not be under a free license. Please satisfy yourself as to
## your rights to use the software. Also, please note that the software in
## multiverse WILL NOT receive any review or updates from the Ubuntu
## security team.
deb http://us.archive.ubuntu.com/ubuntu/ jammy multiverse
# deb-src http://us.archive.ubuntu.com/ubuntu/ jammy multiverse
deb http://us.archive.ubuntu.com/ubuntu/ jammy-updates multiverse
# deb-src http://us.archive.ubuntu.com/ubuntu/ jammy-updates multiverse

## N.B. software from this repository may not have been tested as
## extensively as that contained in the main release, although it includes
## newer versions of some applications which may provide useful features.
## Also, please note that software in backports WILL NOT receive any review
## or updates from the Ubuntu security team.
deb http://us.archive.ubuntu.com/ubuntu/ jammy-backports main restricted universe multiverse
# deb-src http://us.archive.ubuntu.com/ubuntu/ jammy-backports main restricted universe multiverse

deb http://security.ubuntu.com/ubuntu jammy-security main restricted
# deb-src http://security.ubuntu.com/ubuntu jammy-security main restricted
deb http://security.ubuntu.com/ubuntu jammy-security universe
# deb-src http://security.ubuntu.com/ubuntu jammy-security universe
deb http://security.ubuntu.com/ubuntu jammy-security multiverse
# deb-src http://security.ubuntu.com/ubuntu jammy-security multiverse

# This system was installed using small removable media
# (e.g. netinst, live or single CD). The matching "deb cdrom"
# entries were disabled at the end of the installation process.
# For information about how to configure apt package sources,
# see the sources.list(5) manual.
```

Updating sources information:

Copy

```
apt-get update
```

This will check all the sources in the configs and update the information about the latest software available there.

This won't actually *Upgrade* the software. The *Update* will only *Update the information about the packages and not the packages themselves*.

Installing packages

Say you have heard about this amazing terminal multiplexer called `tmux` and you want to give it a try.

Copy

```
$ tmux
The program 'tmux' is currently not installed. You can install it by typing:
sudo apt-get install tmux
$ which tmux
$ type tmux
bash: type: tmux: not found
```

So let's install it. If it's in the repositories it's enough to tell the package manager to install it:

Copy

```
apt-get install tmux
```

Note that

- apt-get install asked for confirmation (Y)
- apt-get resolved *dependencies*, It knows what is needed to install this package and installs them
- Debian packages are something.deb

If you only want a dry-run/simulation:

Copy

```
apt-get install -s tmux
```

and this will only download the files needed into the cache without installing them:

Copy

```
apt-get install --download-only tmux
```

The downloaded packages are stored as a cache in `/var/cache/apt/archive/`.

If you want to download only one specific package, you can do:

Copy

```
apt-get download tmux
```

Removing debian packages

Copy

```
apt-get remove tmux
```

And if you want to remove automatically installed dependencies:

Copy

```
$ apt-get autoremove tmux
```

or even

Copy

```
$ apt-get autoremove
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following packages will be REMOVED:
  linux-image-3.16.0-25-generic linux-image-extra-3.16.0-25-generic
0 upgraded, 0 newly installed, 2 to remove, and 0 not upgraded.
After this operation, 203 MB of disk space will be freed.
Do you want to continue? [Y/n] y
```

To autoremove whatever is not needed anymore.

Notes:

- Removing a package will not remove its dependencies
- If removing a dependency, you'll get a warning about what will be removed alongside this package

Searching for packages

If you are using the apt suit, the search is done via `apt-cache` or you can use the general `apt`.

Copy

```
$ apt-cache search "tiny window"
$ apt search grub2
```

Upgrading

For updating a single package:

Copy

```
apt-get install tzdata
```

And for upgrading whatever is installed:

Copy

```
apt-get upgrade
```

Or going to a new distribution:

Copy

```
apt-get dist-upgrade
```

Note: like most other tools, you can configure the default configs at `/etc/apt/apt.conf` and there is a program apt-config for this purpose.

Reconfiguring packages

Debian packages can have *configuration actions* that will take after the package is installed. This is done by `debconf`. For example, tzdata will ask you about the timezone settings after you installed it. If you want to *reconfigure* a package that is already installed, you can use the `dpkg-reconfigure`:

Copy

```
 dpkg-reconfigure tzdata
```

Package information with dpkg

The underlying tool to work with `.deb` files is the `dpkg`. It is your to-go tool if you want to do manual actions on a deb package. The general format is:

Copy

```
dpkg [OPTIONS] ACTION PACKAGE
```

Some common actions are:

Switch	Description
-c or --contents	Show the contents of a package
-C or --audit	Search for broken installed packages and propose solutions
--configure	Reconfigure an installed package
-i or --install	Install or Upgrade a package; Wont resolve / install dependencies
-I or --info	Show Info
-l or --list	List all installed packages
-L or --listfiles	List all files related to this package
-P or --purge	Remove the package and its configuration files
-r or --remove	Remove the package; Keep the configurations
-s or --status	Display status of a package
-S or --search	Search and see which package owns this file

You can check the contents:

Copy

```
jadi@lpicjadi:/tmp$ dpkg --contents bzr_2.7.0+bzr6622+brz_all.deb
drwxr-xr-x root/root         0 2019-09-19 18:25 ./
drwxr-xr-x root/root         0 2019-09-19 18:25 ./usr/
drwxr-xr-x root/root         0 2019-09-19 18:25 ./usr/share/
drwxr-xr-x root/root         0 2019-09-19 18:25 ./usr/share/doc/
drwxr-xr-x root/root         0 2019-09-19 18:25 ./usr/share/doc/bzr/
-rw-r--r-- root/root       404 2019-09-19 18:25 ./usr/share/doc/bzr/NEWS.Debian.gz
-rw-r--r-- root/root      1301 2019-09-19 18:25 ./usr/share/doc/bzr/changelog.gz
-rw-r--r-- root/root      1769 2019-09-19 18:25 ./usr/share/doc/bzr/copyright
```

Or install a deb package (without its dependencies) or check its status:

Copy

```
$ dpkg -s bzr
Package: bzr
Status: deinstall ok config-files
Priority: optional
Section: vcs
Installed-Size: 102
Maintainer: Ubuntu Developers <ubuntu-devel-discuss@lists.ubuntu.com>
Architecture: all
Version: 2.6.0+bzr6595-1ubuntu1
Config-Version: 2.6.0+bzr6595-1ubuntu1
Depends: python-bzrlib (<= 2.6.0+bzr6595-1ubuntu1.1~), python-bzrlib (>= 2.6.0+bzr6595-1ubuntu1), python:any
Recommends: python-gpgme
Suggests: bzr-doc, bzrtools, python-bzrlib.tests
Breaks: bzr-pqm (<< 1.4.0~bzr80), bzr-xmloutput (<< 0.8.8+bzr160), python-bzrlib (<< 2.4.0~beta3~)
Conffiles:
 /etc/bash_completion.d/bzr b8d9ca95521a7c5f14860e205a854da2
Description: easy to use distributed version control system
 Bazaar is a distributed version control system designed to be easy to
 use and intuitive, able to adapt to many workflows, reliable, and
 easily extendable.
 .
 Publishing of branches can be done over plain HTTP, that is, no special
 software is needed on the server to host Bazaar branches. Branches can
 be pushed to the server via sftp (which most SSH installations come
 with), FTP, or over a custom and faster protocol if bzr is installed in
 the remote end.
 .
 Merging in Bazaar is easy, as the implementation can avoid many
 spurious conflicts deals well with repeated merges between branches,
 and can handle modifications to renamed files correctly.
 .
 Bazaar is written in Python and has a flexible plugin interface that
 can be used to extend its functionality. Many plugins exist, providing
 useful commands (bzrtools), graphical interfaces (qbzr), or native
 interaction with Subversion branches (bzr-svn).
 .
 Install python-paramiko if you are going to push branches to remote
 hosts with sftp, and python-pycurl if you'd like SSL certificates
 always to be verified.
Homepage: http://bazaar-vcs.org
Original-Maintainer: Debian Bazaar Maintainers <pkg-bazaar-maint@lists.alioth.debian.org>
```

Or even **purge** an installed package; Removing the package and all of its configurations. To Purge a package, use the **-P** or **--purge** switch.

There is also **-L** to check the files and directories a package installed:

Copy

```
$ dpkg -L bzr
/.
/usr
/usr/bin
/usr/bin/jcal
/usr/share
/usr/share/doc
/usr/share/doc/jcal
/usr/share/doc/jcal/README
/usr/share/doc/jcal/copyright
/usr/share/man
/usr/share/man/man1
/usr/share/man/man1/jcal.1.gz
/usr/share/doc/jcal/changelog.Debian.gz
```

and **-S** will show which package installed the given file:

Copy

```
$ dpkg -S /var/lib/mplayer/prefs/mirrors
mplayer: /var/lib/mplayer/prefs/mirrors
```

Common apt-get options

Option	Usage
autoclean	Removes unused packages
check	Check db for issues
clean	Clean the DB, you can do a `clean all` to clean everything and start afresh
dist-upgrade	Checks for new versions of the OS; Major upgrade
install	Install or upgrade packages
remove	Removes a package
source	Install the source of a package
update	Updates the information about packages from repositories
upgrade	Upgrades all packages

> In some cases a package is installed but without proper dependencies (say using `dpkg`) or an installation is interrupted for any reason. In these cases a `apt-get install -f` might help, `-f` is for `fix broken`.

Common apt-cache options

Option	Usage
depends	Show dependencies
pkgnames	Shows all installed packages
search	Search
showpkg	Show information about a package
stats	Show statistics
unmet	Show unmet dependencies for all installed packages or the one you specified

Other tools

There are even more tools, the tools with fancy GUIs or text-based tools and user interface tools like `aptitude`.

102.5 Use RPM and YUM package management

Weight: 3

Candidates should be able to perform package management using RPM, YUM, and Zypper.

Key Knowledge Areas

- Install, re-install, upgrade and remove packages using RPM, YUM, and Zypper
- Obtain information on RPM packages such as version, status, dependencies, integrity, and signatures Determine what files a package provides, as well as find which package a specific file comes from Awareness of dnf

The following is a partial list of the used files, terms, and utilities:

Terms and Utilities

- rpm
- rpm2cpio
- `/etc/yum.conf`
- `/etc/yum.repos.d/`
- yum
- zypper

Introduction

RedHat Package Manager (RPM) and **YellowDog Update Manager (YUM)** are used by Fedora, RedHat, RHEL, CentOS, RocksOS, ... to manage packages. The package format is called RPM and can be managed by `rpm` tools but if you want to use the repositories to install, update, search, ... packages, or even upgrade the whole system, you can use the `yum` command. To have a deeper understanding of the repositories, please refer to the previous section (102.4); Here I assume that you know the concept.

yum

`yum` is the package manager used by RedHat-based systems. Its configuration files are located at `/etc/yum.conf` and `/etc/yum.repos.d/`. Below is a sample.

Copy

```
# cat /etc/yum.conf
[main]
cachedir=/var/cache/yum/$basearch/$releasever
keepcache=0
debuglevel=2
logfile=/var/log/yum.log
exactarch=1
obsoletes=1
gpgcheck=1
plugins=1
installonly_limit=3
```

And here is a sample of an actual Repo file on a Fedora system:

Copy

```
# cat /etc/yum.repos.d/fedora.repo
[fedora]
name=Fedora $releasever - $basearch
#baseurl=http://download.example/pub/fedora/linux/releases/$releasever/Everything/$basearch/os/
metalink=https://mirrors.fedoraproject.org/metalink?repo=fedora-$releasever&arch=$basearch
```

```
enabled=1
countme=1
metadata_expire=7d
repo_gpgcheck=0
type=rpm
gpgcheck=1
gpgkey=file:///etc/pki/rpm-gpg/RPM-GPG-KEY-fedora-$releasever-$basearch
skip_if_unavailable=False

[fedora-debuginfo]
name=Fedora $releasever - $basearch - Debug
#baseurl=http://download.example/pub/fedora/linux/releases/$releasever/Everything/$basearch/debug/tree/
metalink=https://mirrors.fedoraproject.org/metalink?repo=fedora-debug-$releasever&arch=$basearch
enabled=0
metadata_expire=7d
repo_gpgcheck=0
type=rpm
gpgcheck=1
gpgkey=file:///etc/pki/rpm-gpg/RPM-GPG-KEY-fedora-$releasever-$basearch
skip_if_unavailable=False

[fedora-source]
name=Fedora $releasever - Source
#baseurl=http://download.example/pub/fedora/linux/releases/$releasever/Everything/source/tree/
metalink=https://mirrors.fedoraproject.org/metalink?repo=fedora-source-$releasever&arch=$basearch
enabled=0
metadata_expire=7d
repo_gpgcheck=0
type=rpm
gpgcheck=1
gpgkey=file:///etc/pki/rpm-gpg/RPM-GPG-KEY-fedora-$releasever-$basearch
skip_if_unavailable=False
```

We use yum like `yum [OPTIONS] [COMMAND] [PACKAGE_NAME]`.

One of the most important options is `-y` which says *yes* to the Y/N questions.

And here you can find some of the commands:

Command	Description
update	Updates the repositories and update the names of packages, or all if nothing is named
install	Install a package
reinstall	Reinstall a package
list	Show a list of packages
info	Show information about a package
remove	Removes an installed package
search	Searches repositories for packages
provides	Check which packages provide a specific file
upgrade	Upgrades packages and removes the obsolete ones
localinstall	Install from a local rpm file
localupdate	Updates from a local rpm file
check-update	Checks repositories for updates to the installed packages
deplist	Shows dependencies of a package
groupinstall	Install a group, say "KDE Plasma Workspaces"
history	Show history of the usage

This is a sample installation:

Copy

```
# yum install bzr
Last metadata expiration check: 0:00:47 ago on Tue 21 Jun 2022 06:38:00 PM +0430.
Dependencies resolved.
================================================================================================================================
 Package                                Architecture       Version                        Repository            Size
================================================================================================================================
Installing:
 breezy                                 x86_64             3.2.1-3.fc36                   fedora               6.0 M
Installing dependencies:
 libsodium                              x86_64             1.0.18-9.fc36                  fedora               163 k
 python3-bcrypt                         x86_64             3.2.2-1.fc36                   updates               43 k
 python3-certifi                        noarch             2021.10.8-1.fc36               fedora                15 k
 python3-configobj                      noarch             5.0.6-27.fc36                  fedora                63 k
 python3-cryptography                   x86_64             36.0.0-3.fc36                  fedora               1.0 M
```

```
 python3-dulwich                    x86_64          0.20.32-1.fc36              fedora          408 k
 python3-httplib2                   noarch          0.20.3-2.fc36               fedora          122 k
 python3-jeepney                    noarch          0.7.1-2.fc36                fedora          324 k
 python3-jwt                        noarch          2.4.0-1.fc36                updates          41 k
 python3-keyring                    noarch          23.6.0-1.fc36               updates          78 k
 python3-lazr-restfulclient         noarch          0.14.4-2.fc36               fedora           84 k
 python3-lazr-uri                   noarch          1.0.6-2.fc36                fedora           33 k
 python3-oauthlib                   noarch          3.0.2-12.fc36               fedora          169 k
 python3-paramiko                   noarch          2.11.0-1.fc36               updates         303 k
 python3-patiencediff               x86_64          0.2.2-4.fc36                fedora           45 k
 python3-pynacl                     x86_64          1.4.0-5.fc36                fedora          108 k
 python3-secretstorage              noarch          3.3.1-4.fc36                fedora           35 k
 python3-wadllib                    noarch          1.3.6-2.fc36                fedora           60 k
Installing weak dependencies:
 python3-jwt+crypto                 noarch          2.4.0-1.fc36                updates         8.9 k
 python3-launchpadlib               noarch          1.10.15.1-2.fc36            fedora          167 k
 python3-oauthlib+signedtoken       noarch          3.0.2-12.fc36               fedora          8.5 k
 python3-pyasn1                     noarch          0.4.8-8.fc36                fedora          134 k

Transaction Summary
================================================================================================================================
Install  23 Packages

Total download size: 9.4 M
Installed size: 44 M
Is this ok [y/N]: y
Downloading Packages:
(1/23): python3-certifi-2021.10.8-1.fc36.noarch.rpm                          1.8 kB/s |  15 kB     00:08
(2/23): libsodium-1.0.18-9.fc36.x86_64.rpm                                    15 kB/s | 163 kB     00:10
(3/23): python3-configobj-5.0.6-27.fc36.noarch.rpm                            10 kB/s |  63 kB     00:06
(4/23): breezy-3.2.1-3.fc36.x86_64.rpm                                       262 kB/s | 6.0 MB     00:23
(5/23): python3-dulwich-0.20.32-1.fc36.x86_64.rpm                             47 kB/s | 408 kB     00:08
(6/23): python3-cryptography-36.0.0-3.fc36.x86_64.rpm                         77 kB/s | 1.0 MB     00:13
(7/23): python3-httplib2-0.20.3-2.fc36.noarch.rpm                            105 kB/s | 122 kB     00:01
(8/23): python3-jeepney-0.7.1-2.fc36.noarch.rpm                              259 kB/s | 324 kB     00:01
(9/23): python3-launchpadlib-1.10.15.1-2.fc36.noarch.rpm                      74 kB/s | 167 kB     00:02
(10/23): python3-lazr-restfulclient-0.14.4-2.fc36.noarch.rpm                  36 kB/s |  84 kB     00:02
(11/23): python3-lazr-uri-1.0.6-2.fc36.noarch.rpm                             15 kB/s |  33 kB     00:02
(12/23): python3-oauthlib+signedtoken-3.0.2-12.fc36.noarch.rpm               4.2 kB/s | 8.5 kB     00:02
(13/23): python3-oauthlib-3.0.2-12.fc36.noarch.rpm                            58 kB/s | 169 kB     00:02
(14/23): python3-patiencediff-0.2.2-4.fc36.x86_64.rpm                         15 kB/s |  45 kB     00:02
(15/23): python3-pyasn1-0.4.8-8.fc36.noarch.rpm                               61 kB/s | 134 kB     00:02
(16/23): python3-pynacl-1.4.0-5.fc36.x86_64.rpm                               36 kB/s | 108 kB     00:03
(17/23): python3-secretstorage-3.3.1-4.fc36.noarch.rpm                        12 kB/s |  35 kB     00:02
(18/23): python3-wadllib-1.3.6-2.fc36.noarch.rpm                              24 kB/s |  60 kB     00:02
(19/23): python3-bcrypt-3.2.2-1.fc36.x86_64.rpm                               16 kB/s |  43 kB     00:02
(20/23): python3-jwt+crypto-2.4.0-1.fc36.noarch.rpm                          3.2 kB/s | 8.9 kB     00:02
(21/23): python3-jwt-2.4.0-1.fc36.noarch.rpm                                  16 kB/s |  41 kB     00:02
(22/23): python3-keyring-23.6.0-1.fc36.noarch.rpm                             18 kB/s |  78 kB     00:04
(23/23): python3-paramiko-2.11.0-1.fc36.noarch.rpm                            38 kB/s | 303 kB     00:08
--------------------------------------------------------------------------------------------------------------------------------
Total                                                                        177 kB/s | 9.4 MB     00:54
Running transaction check
Transaction check succeeded.
Running transaction test
Transaction test succeeded.
Running transaction
  Preparing        :                                                                                                        1/1
  Installing       : python3-cryptography-36.0.0-3.fc36.x86_64                                                              1/23
  Installing       : python3-lazr-uri-1.0.6-2.fc36.noarch                                                                   2/23
  Installing       : python3-jeepney-0.7.1-2.fc36.noarch                                                                    3/23
  Installing       : python3-httplib2-0.20.3-2.fc36.noarch                                                                  4/23
  Installing       : python3-secretstorage-3.3.1-4.fc36.noarch                                                              5/23
  Installing       : python3-keyring-23.6.0-1.fc36.noarch                                                                   6/23
  Installing       : python3-wadllib-1.3.6-2.fc36.noarch                                                                    7/23
  Installing       : python3-jwt-2.4.0-1.fc36.noarch                                                                        8/23
  Installing       : python3-jwt+crypto-2.4.0-1.fc36.noarch                                                                 9/23
  Installing       : python3-oauthlib-3.0.2-12.fc36.noarch                                                                 10/23
  Installing       : python3-oauthlib+signedtoken-3.0.2-12.fc36.noarch                                                     11/23
  Installing       : python3-lazr-restfulclient-0.14.4-2.fc36.noarch                                                       12/23
  Installing       : python3-launchpadlib-1.10.15.1-2.fc36.noarch                                                          13/23
  Installing       : python3-bcrypt-3.2.2-1.fc36.x86_64                                                                    14/23
  Installing       : python3-pyasn1-0.4.8-8.fc36.noarch                                                                    15/23
  Installing       : python3-patiencediff-0.2.2-4.fc36.x86_64                                                              16/23
  Installing       : python3-configobj-5.0.6-27.fc36.noarch                                                                17/23
  Installing       : python3-certifi-2021.10.8-1.fc36.noarch                                                               18/23
  Installing       : python3-dulwich-0.20.32-1.fc36.x86_64                                                                 19/23
  Installing       : libsodium-1.0.18-9.fc36.x86_64                                                                        20/23
  Installing       : python3-pynacl-1.4.0-5.fc36.x86_64                                                                    21/23
  Installing       : python3-paramiko-2.11.0-1.fc36.noarch                                                                 22/23
  Installing       : breezy-3.2.1-3.fc36.x86_64                                                                            23/23
```

```
  Running scriptlet: breezy-3.2.1-3.fc36.x86_64                                          23/23
  Verifying        : breezy-3.2.1-3.fc36.x86_64                                           1/23
  Verifying        : libsodium-1.0.18-9.fc36.x86_64                                       2/23
  Verifying        : python3-certifi-2021.10.8-1.fc36.noarch                              3/23
  Verifying        : python3-configobj-5.0.6-27.fc36.noarch                               4/23
  Verifying        : python3-cryptography-36.0.0-3.fc36.x86_64                            5/23
  Verifying        : python3-dulwich-0.20.32-1.fc36.x86_64                                6/23
  Verifying        : python3-httplib2-0.20.3-2.fc36.noarch                                7/23
  Verifying        : python3-jeepney-0.7.1-2.fc36.noarch                                  8/23
  Verifying        : python3-launchpadlib-1.10.15.1-2.fc36.noarch                         9/23
  Verifying        : python3-lazr-restfulclient-0.14.4-2.fc36.noarch                     10/23
  Verifying        : python3-lazr-uri-1.0.6-2.fc36.noarch                                11/23
  Verifying        : python3-oauthlib+signedtoken-3.0.2-12.fc36.noarch                   12/23
  Verifying        : python3-oauthlib-3.0.2-12.fc36.noarch                               13/23
  Verifying        : python3-patiencediff-0.2.2-4.fc36.x86_64                            14/23
  Verifying        : python3-pyasn1-0.4.8-8.fc36.noarch                                  15/23
  Verifying        : python3-pynacl-1.4.0-5.fc36.x86_64                                  16/23
  Verifying        : python3-secretstorage-3.3.1-4.fc36.noarch                           17/23
  Verifying        : python3-wadllib-1.3.6-2.fc36.noarch                                 18/23
  Verifying        : python3-bcrypt-3.2.2-1.fc36.x86_64                                  19/23
  Verifying        : python3-jwt+crypto-2.4.0-1.fc36.noarch                              20/23
  Verifying        : python3-jwt-2.4.0-1.fc36.noarch                                     21/23
  Verifying        : python3-keyring-23.6.0-1.fc36.noarch                                22/23
  Verifying        : python3-paramiko-2.11.0-1.fc36.noarch                               23/23

Installed:
  breezy-3.2.1-3.fc36.x86_64                               libsodium-1.0.18-9.fc36.x86_64
  python3-bcrypt-3.2.2-1.fc36.x86_64                       python3-certifi-2021.10.8-1.fc36.noarch
  python3-configobj-5.0.6-27.fc36.noarch                   python3-cryptography-36.0.0-3.fc36.x86_64
  python3-dulwich-0.20.32-1.fc36.x86_64                    python3-httplib2-0.20.3-2.fc36.noarch
  python3-jeepney-0.7.1-2.fc36.noarch                      python3-jwt-2.4.0-1.fc36.noarch
  python3-jwt+crypto-2.4.0-1.fc36.noarch                   python3-keyring-23.6.0-1.fc36.noarch
  python3-launchpadlib-1.10.15.1-2.fc36.noarch             python3-lazr-restfulclient-0.14.4-2.fc36.noarch
  python3-lazr-uri-1.0.6-2.fc36.noarch                     python3-oauthlib-3.0.2-12.fc36.noarch
  python3-oauthlib+signedtoken-3.0.2-12.fc36.noarch        python3-paramiko-2.11.0-1.fc36.noarch
  python3-patiencediff-0.2.2-4.fc36.x86_64                 python3-pyasn1-0.4.8-8.fc36.noarch
  python3-pynacl-1.4.0-5.fc36.x86_64                       python3-secretstorage-3.3.1-4.fc36.noarch
  python3-wadllib-1.3.6-2.fc36.noarch

Complete!
```

You can also use the wildcards:

Copy

```
# yum update 'cal*'
```

> Fun fact: Fedora Linux uses `dnf` as its package manager and will translate your `yum` commands to its `dnf` equivalents.

yumdownloader

This tool will download rpms from repositories without installing them. If you need to download all the dependencies too, use the `--resolve` switch:

Copy

```
yumdownloader --resolve bzr
```

RPM

The `rpm` command can run ACTIONs on individual RPM files. You can use it like `rpm ACTION [OPTION] rpm_file.rpm`

One of common options is `-v` for verbose output and these are the common ACTIONs:

Short Form	Long Form	Description
-i	--install	Installs a package
-e	--erase	Removes a package
-U	--upgrade	Installs/Upgrades a package
-q	--query	Checks if the package is installed
-F	--freshen	Only update if it's already installed
-V	--verify	Check the integrity of the installation
-K	--checksig	Checks the integrity of an rpm package

Please note that each action might have its specific options.

Install and update

In most cases, we use `-U` which Installs or upgrades a package.

- RPM does not have a database of automatic package installation, so it can not remove the automatically installed dependencies.

If you have an rpm with all of its dependencies, you can install them using `rpm -Uvh *.rpm`. This will tell rpm not to complain about the dependencies if it is presented in other files. Here the `-h` creates 50 hash signs to show the progress.

In some cases - if you know what you are doing - you can use `--nodeps` to prevent the dependency check or even use `--force` to force the install / upgrade despite all the issues & complains.

Query

A normal query is like this:

Copy

```
[root@fedora tmp]# rpm -q breezy-3.2.1-3.fc36.x86_64.rpm
breezy-3.2.1-3.fc36.x86_64
[root@fedora tmp]# rpm -q breezy
breezy-3.2.1-3.fc36.x86_64
[root@fedora tmp]# rpm -q emacs
package emacs is not installed
```

And you can use these options to spice it up:

Short	Long	Description
-c	--configfiles	Show the packages configuration files
-i	--info	Detailed info about a pacakge
-a	--all	Show all Installed packages
	--whatprovides	Shows what packages provide this file
-l	--list	Query the list of files a package installs
-R	--requires	Show dependencies of a package
-f	--file	Query package owning file

Verify

You can verify your packages and see if they are installed correctly or not. You can use the `-Vv` option for verbose output or just use the `-V` to verify and see only the issues. This is the output after I edited the `/bin/tmux` manually:

Copy

```
[root@fedora tmp]# rpm -V tmux
S.5....T.    /usr/bin/tmux
```

And this is part of the `man rpm`'s `-V` section:

Copy

```
    S Size differs
    M Mode differs (includes permissions and file type)
    5 digest (formerly MD5 sum) differs
    D Device major/minor number mismatch
    L readLink(2) path mismatch
    U User ownership differs
    G Group ownership differs
    T mTime differs
    P caPabilities differ
```

You can also check the integrity of an rpm package with `-K`:

Copy

```
# rpm -Kv breezy-3.2.1-3.fc36.x86_64.rpm
breezy-3.2.1-3.fc36.x86_64.rpm:
    Header V4 RSA/SHA256 Signature, key ID 38ab71f4: OK
    Header SHA256 digest: OK
    Header SHA1 digest: OK
    Payload SHA256 digest: OK
    V4 RSA/SHA256 Signature, key ID 38ab71f4: OK
    MD5 digest: OK
```

The above output shows that this file is valid.

Uninstall

Copy

```
[root@fedora tmp]# rpm -e tmux
error: Failed dependencies:
    tmux is needed by (installed) anaconda-install-env-deps-36.16.5-1.fc36.x86_64
```

- rpm removes the package without asking!
- rpm won't remove a package that is needed by another package

Extract RPM Files

rpm2cpio

The **cpio** is an archive format (Just like zip or rar or tar). You can use the `rpm2cpio` command to convert RPM files to *cpio* and then use the `cpio` tool to extract them:

Copy

```
[root@fedora tmp]# rpm2cpio breezy-3.2.1-3.fc36.x86_64.rpm > breezy.cpio
[root@fedora tmp]# cpio -idv < breezy.cpio
./usr/bin/brz
./usr/bin/bzr
./usr/bin/bzr-receive-pack
./usr/bin/bzr-upload-pack
./usr/bin/git-remote-brz
./usr/bin/git-remote-bzr
[...]
```

Zypper

The SUSE Linux and its sibling openSUSE use ZYpp as their package manager engine. You can use YAST or Zypper tools to communicate with it.

These are the main commands used in `zypper`:

Command	Description
help	General help
install	Installs a package
info	Displays information of a package
list-updates	Shows available updates
lr	Shows repository information
packages	List all available packages or packages from a specific repo
what-provides	Show the owner of a file
refresh	Refreshes the repositories information
remove	Removes a package from the system
search	Searches for a package
update	Checks the repositories and updates the installed packages
verify	Checks a package and its dependencies

> You can shorten the command when using `zypper`, so `zypper se tmux` will *search* for tmux.

Other tools

YUM and RPM are the main package managers on Fedora, RHEL & Centos but other tools are also available. As mentioned, SUSE uses `YaST`, and some modern desktops (KDE & Gnome) use `PackageKit` which is a graphical tool. It is also good to note that the `dnf` suite is also gaining popularity and is pre-installed on Fedora systems.

102.6 Linux as a virtualization guest

Weight: 1

Candidates should understand the implications of virtualization and cloud computing on a Linux guest system.

Key Knowledge Areas

- Understand the general concept of virtual machines and containers
- Understand common elements of virtual machines in an IaaS cloud, such as computing instances, block storage, and networking
- Understand the unique properties of a Linux system which have to be changed when a system is cloned or used as a template
- Understand how system images are used to deploy virtual machines, cloud instances, and containers
- Understand Linux extensions that integrate Linux with a virtualization product. Awareness of cloud-init

Terms

- Virtual machine
- Linux container
- Application container
- Guest drivers
- SSH host keys
- D-Bus machine id

Introduction

Virtual machines (VMs) are *simulated* computers. You can use Virtual Machines to create new computers on top of your running machine and install new OSs there. In some cases, it's also possible to run only parts of an OS on top of the current OS; This is called having containers.

To run virtual machines, we need **Hypervisor** software (also called Virtual Machine Managers (VMM)). We have two types of hypervisors.

To check and see if your host operating system / CPU, supports using hypervisors check for the `vmx` (for Intel CPUs) or `svm` (for AMD CPUs) in your `/proc/cpuinfo` in flags.

> You may need to turn the hypervisor option On using your BIOS or UEFI.

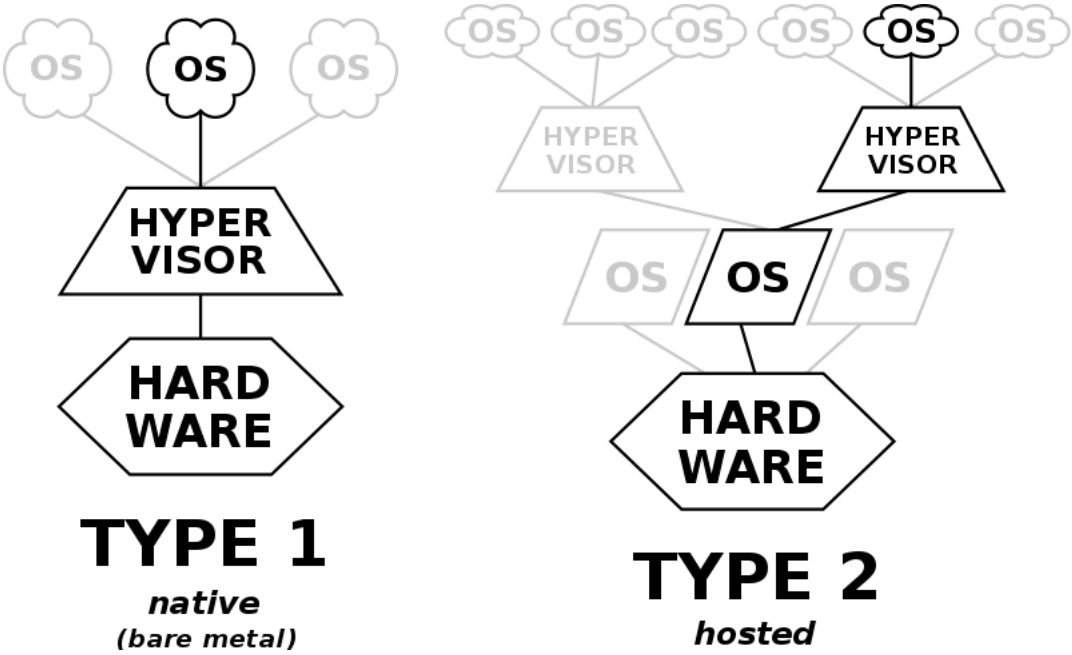
Based on your CPU you should have `kvm` or `kvm-amd` kernel modules loaded.

Copy

```
lsmod | grep -i kvm
sudo modprobe kvm
```

> If you see `hypervisor` in your `/proc/cpuinfo` it means that you are inside a virtualized Linux machine :)

Now, let's see the 2 types of hypervisors. First type 2, since it's easier to understand.



Type 2 Hypervisor

These hypervisors run on a conventional operating system (OS) just as other computer programs do. A guest operating system runs as a process on the host. Type-2 hypervisors abstract guest operating systems from the host operating system.

In other words, a type 2 hypervisor is the software between the guest and host. It completely runs on the host OS and provides virtualization to the guest.

Two of the most famous Type 2 hypervisors are VirtualBox (from Oracle) and VMware.

Type 1 Hypervisor

These hypervisors run directly on the host's hardware to control the hardware and manage guest operating systems. For this reason, they are sometimes called bare-metal hypervisors. The first hypervisors, which IBM developed in the 1960s, were native hypervisors. These included the test software SIMMON and the CP/CMS operating system, the predecessor of IBM z/VM.

Some of the most famous Type 1 hypervisors are KVM, Xen & Hyper-V. **KVM** is built-in since Linux Kernel version 2.6.20.

Creating a Virtual Machine

First, create the machine itself. We tell the hypervisor this machine how much RAM/disk/CPU/... needs and set a name for our machine. Then we need to *install* the guest OS. This can be done using:

- Installing from a CD / DVD / ...
- **Cloning** an existing machine
- Using **Open Virtualization Format (OVF)** to move machines between hypervisors. This is a standard format for virtual machine definition and may include several files, in this case, you can archive all of them into one **Open Virtualization Archive (OVA)** file.
- It is also possible to create **Templates** that are *master copies** to initiate new machines

  You may need to install some *guest drivers* or *additions* to help your hypervisor to have better control over your guest machine. These might include graphical drivers for VirtualBox or scripts to help VMWare to control a guest machine or check its status.

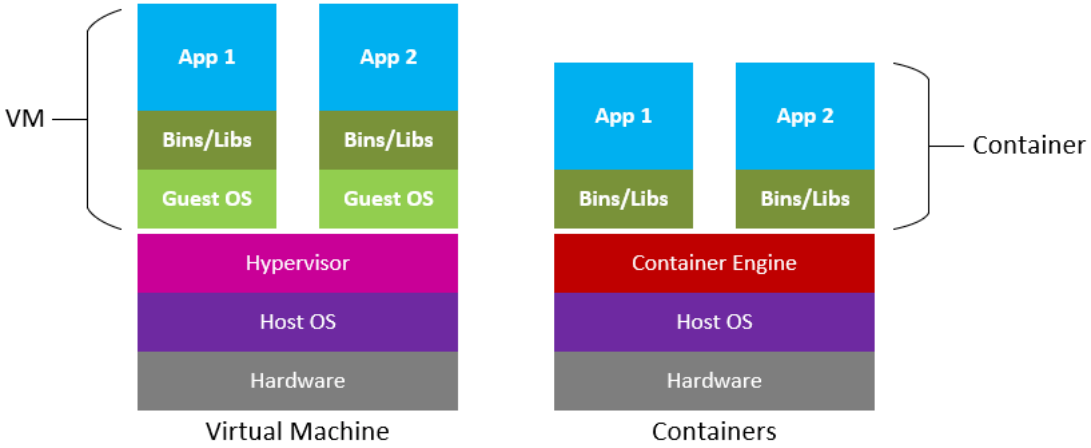
Guest-specific configs

Some configurations are machine specific. For example, a network card's MAC address should be unique for whole the network. If we are cloning a machine or sometimes creating them from templates, at least we need to change these on each machine before booting them:

- Host Name
- NIC MAC Address
- NIC IP (If not using DHCP)
- Machine ID (delete the `/etc/machine-id` and `/var/lib/dbus/machine-id` and run `dbus-uuidgen --ensure`. These two files might be soft links to each other)
- Encryption Keys like SSH Fingerprints and PGP keys
- HDD UUIDs
- Any other UUIDs on the system

  Some configs might be empty on templates, do not forget to fill them too.

Containers



In previous sections, we were dealing with complete guest OSs. But is also possible to virtualize only parts of an OS; This is called OS-level virtualization.

OS-level virtualization is an operating system (OS) paradigm in which the kernel allows the existence of multiple isolated user-space instances, called containers.

This can be done to run an application or a service or even run most parts of a new OS for test purposes.

IaaS

As the name implies, **Infrastructure as a Service** or IaaS means offloading parts of your infrastructure to other companies. This means buying *services* like electricity, cooling, and even running hypervisors to another company and just renting your VirtualMachine from them. This makes your life easier because things like "Adding New Hards" now only mean paying a bit more for more HDD on your machine; Instead of buying HDDs and installing them, ... This is called cloud! You might even be able to move your machine from one continent to another one just with one click.

Samples of these cloud providers are Amazon Web Services, Google Cloud Platform, and Microsoft Azure.

Different cloud providers might provide different levels of Infrastructure or services. These are some examples:

- Load Balancing: Distribute incoming requests between your servers
- Block Storage: Providing disks to be configured by you and added to your machines
- Object Storage: Lets you store your data directly; Say for photos
- Elasticity: Lets you configure an automated increase/decrease in your service capacity based on request volume
- SaaS: Software as a Service lets you use the software you need, on the Cloud as a Service. Think of having an online office suite for your company without installing anything on your workstations.

There are programs like `cloud-init` which help you initialize your cloud machine with ease. This service can help you start machines based on templates on AWS, Azure, Digital Ocean, and others with ease.

103.1 Work on the command line

Weight: 4

Description: Candidates should be able to interact with shells and commands using the command line. The objective assumes the Bash shell.

Objectives

- Use single shell commands and one-line command sequences to perform basic tasks on the command line.
- Use and modify the shell environment including defining, referencing, and exporting environment variables.
- Use and edit command history.
- Invoke commands inside and outside the defined path.

Terms

- bash
- echo
- env
- export
- pwd
- set
- unset
- type
- which
- man
- uname
- history
- `.bash_history`
- Quoting

Shells and Bash

You issue your commands in a shell; it's your command line interface and you have various options for it. To reach your shell you should login into the system in the text mode or run one of the various *Terminal Emulators* in your GUI. Some samples are `gnome-terminal`, `konsole`, `xterm`, etc.

After running the Terminal Emulator or logging into the text mode, you are in the shell and you can issue commands. Although `bash` (GNU Bourne Again shell) is the most common one, you might use `zsh`, `dash`, `ksh`, `csh`, and others.

You can check where your general `sh` command links to via

Copy

```
$ readlink /bin/sh
```

Or check your `$SHELL` variable using:

Copy

```
echo $SHELL
```

Your bash has some *internal* commands that it understands without any external dependency (say `cd`, `break`, `exec`, ...) but if it does not understand something internally, it will try to run it as an external executable.

You can use the `type` command to determine this:

Copy

```
[jadi@fedora ~]$ type cd
cd is a shell builtin
[jadi@fedora ~]$ type ls
ls is aliased to `ls --color=auto'
[jadi@fedora ~]$ type ping
ping is /usr/bin/ping
```

cd, pwd & uname

cd

You've already seen lots of `cd` commands :) it *changes directory*, including `.` (current directory) and `..` (parent directory).

You can point to directories in two ways:

1. **Absolute Paths**: Like `/home/jadi/lpic1/lesson3.1`
2. **Relative Paths**: Like `lpic1/lesson3.1`. In this case, we are not adding the `/` in the beginning so the bash will try to find `lpic1` directory where we are (local / relative)

   The `~` characters means *home directory of the user issuing the command*

It is also possible to issue the `cd` without any parameters. It will move you to your home directory. So these 3 commands are all equal:

Copy

```
cd
cd ~
cd $HOME
```

pwd

Will show you your current directory:

Copy

```
[jadi@fedora lesson3.1]$ pwd
/home/jadi/lpic1/lesson3.1
```

uname

Gives you data about the system. Common switches are:

Option	Description
-s	Print the kernel name. This is the default if no option is specified.
-n	Print the nodename or hostname.
-r	Print the release of the kernel. This option is often used with module-handling commands.
-v	Print the version of the kernel.
-m	Print the machine's hardware (CPU) name.
-o	Print the operating system name.
-a	Print all of the above information.

Example:

Copy

```
[jadi@fedora lesson3.1]$ uname -a
Linux fedora 5.14.0-60.fc35.aarch64 #1 SMP Mon Aug 30 16:30:42 UTC 2021 aarch64 aarch64 aarch64 GNU/Linux
```

Getting Help

Most of the commands we use do have a cool and complete manual, accessible via the `man` command. It uses the `less` pager by default and contains the documentation, switches, parameters, ... of commands, and utilities.

Make yourself familiar with the man by reading the manual of the `yes` command:

Copy

```
$ man yes
```

Please note that man pages are categorized in different sections (books). You can check these by reading the man's manual:

Copy

```
$ man man
$ man 5 passwd
```

`

Special characters and Quoting/Escaping

In the computer world, some characters do have special meanings. For example in bash, the `*` character will expand to all files. In these cases, if you want to use this character without this expansion, you have to *Quote* it or *Escape* it. In many cases this is done via adding a `\` character before it:

Copy

```
$ echo 2 \* 3 = 6
2 * 3 = 6
```

These are the character with special meanings that you need to quote if you are using them in your commands:

```
* ?[]'"\$;&()|^<>
```

Please note that there is a space character in the character list above.

> As you can see, the `\` has a specific meaning so if you want to use the back-slash itself (without its escaping usage), you have to *quote* your back-slash with another back-slash `\\`.

In addition to escaping, you can use `\` to create some special characters. For example, as you can not type a *return* character, you create it via `\n` (new line):

Copy

```
jadi@funlife:~$ echo -e "hello\nthere"
hello
there
```

Some other cases are:

Escape sequence	Function
\a	Alert (bell)
\b	Backspace
\c	Suppress trailing newline (same function as -n option)
\f	Form feed (clear the screen on a video display)
\n	New line
\r	Carriage return
\t	Horizontal tab

On bash you can use `\` to break a command into more lines:

Copy

```
$ echo You know slashes! But this \
is another \
usage
You know slashes! But this is another usage
```

Shell environment variables

Environment Variables contain some configs and information about the shell. For example, your default editor is set in the `EDITOR` variable. You can query the value of a shell variable like this:

Copy

```
[jadi@fedora ~]$ echo $EDITOR
/usr/bin/nano
```

It is possible to check all the env variables using the `set` or `env` command.

These are some of the most used bash environment variables:

Name	Function
USER	The name of the logged-in user
PATH	List of directories to search for commands, colon separated
EDITOR	Default editor
HISTFILE	Where bash should save its history (normally `.bash_history`)
HOSTNAME	System hostname
PS1	The Prompt! Play with it
UID	The numeric user id of the logged-in user
HOME	The user's home directory
PWD	The current working directory
SHELL	The name of the shell

Name	Function
$	The process id (or PID of the running bash shell (or other) process
PPID	The process id of the process that started this process (that is, the id of the parent process)
?	The exit code of the last command

When trying to access the value, you should add a `$` to the beginning of the variable name.

Copy

```
$ echo $USER $UID
jadi 1000
$ echo $SHELL $HOME $PWD
/bin/bash /home/jadi /home/jadi/lpic
```

To define a new EV (Environment Variable) or change or delete one, we can do:

Copy

```
$ MYMOOD=happy
$ echo I am $MYMOOD
I am happy
$ MYMOOD="Even Happier" # space has a specific meaning
$ unset MM
```

If you want new programs starting from this shell to have access to the variable you defined, you have to set them with `export` or export them later.

Copy

```
$ export MYMOOD
$ export YOURMOOD="Not Confused"
```

> Global bash configs are stored at `/etc/profile` and each user has her config at `~/.profile` & `~/.bash\_profile` & `~/.bash\_logout`. If you need a permanent change, add your configs to these.

Path

When you issue a command, bash will run if it's an internal bash command. Otherwise, bash will go and check the `PATH` variables one by one and will try to find them there. If not, it will give you an error. If you want to run something on a specific path, you have to exclusively describe the location:

Copy

```
$ echo $PATH
/home/jadi/bin:/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin:/usr/games:/usr/local/games
```

But what happens if I try to run `tar`? Let's check with `which`, `type`, and `whereis` commands:

Copy

```
jadi@funlife:~$ which tar
/bin/tar
jadi@funlife:~$ type tar
tar is /bin/tar
jadi@funlife:~$ whereis tar
tar: /usr/lib/tar /bin/tar /usr/include/tar.h /usr/share/man/man1/tar.1.gz
```

A cooler example is `ping` on Fedora:

Copy

```
[jadi@fedora ~]$ whereis ping
ping: /usr/bin/ping /usr/sbin/ping /usr/share/man/man8/ping.8.gz
[jadi@fedora ~]$ which ping
/usr/bin/ping
[jadi@fedora ~]$ /usr/sbin/ping 4.2.2.4
PING 4.2.2.4 (4.2.2.4) 56(84) bytes of data.
64 bytes from 4.2.2.4: icmp_seq=1 ttl=50 time=160 ms
```

> That's why when you want to say "run this_program in this directory" you issue "./this_program". You are exclusively telling bash where the file is. In Linux, the current directory (.) is not part of the PATH by default.

Command history

Bash saves its history in `~/.bash_history`. You can `cat` it and see its contents or run the `history` command. You can also use the below keys (combinations) to access your previous commands:

Key (Combination)	Usage
Up and Down Arrow	Move in the history
Ctrl+R	Backward Search
Ctrl+O	Run the command you found with Ctrl+R
!!	Run the last command
!10	Run command number 10
!text	search backward for text, and run the first found command

If you want to clear your history, issue `HISTSIZE=0

Exiting the Shell

The `exit` command exits the shell. Same as CTRL+d.

If you run a command inside parentheses that command will be run inside a sub-shell and `exec` will run a command and closes the current shell.

103.2 Process text streams using filters

Weight: 2

Description: Candidates should be able to apply filters to text streams.

Objectives

Send text files and output streams through text utility filters to modify the output using standard UNIX commands found in the GNU textutils package.

Terms

- bzcat
- cat
- cut
- head
- less
- md5sum
- nl
- od
- paste
- sed
- sha256sum
- sha512sum
- sort
- split
- tail
- tr
- uniq
- wc
- xzcat
- zcat

Streams

In **UNIX** world a lot of data is in TEXT form. Log files, configurations, data, etc. **Filtering** this data means taking an input stream of text and performing some conversion on the text before sending it to an output stream. In this context, a **streams** is nothing more than *"a sequence of bytes that can be read or written using library functions that hide the details of an underlying device from the application"*.

In simple words, a text stream is an input of text from a keyboard, a file, a network device, ... which can be viewed, changed, examined, and ... via text util commands.

Modern programming environments and shells (including bash) use three standard I/O streams:

- **stdin** is the standard input stream, which provides input to commands.
- **stdout** is the standard output stream, which displays output from commands.
- **stderr** is the standard error stream, which displays error output from commands

Here we are talking about the **stdin** and viewing or manipulating it via different commands and utilities. You will see more about these streams and will see how we can combine commands to *PIPE* inputs and outputs of different commands in chapter 103.4.

Viewing commands

cat

This command simply outputs its input stream (or the filename you give it). As you saw in the previous section. As with most commands, if you do not give input to it, it will read the data from the keyboard.

Copy

```
jadi@funlife:~/w/lpic/101$ cat > mydata
test
this is the second line
bye
jadi@funlife:~/w/lpic/101$ cat mydata
test
this is the second line
bye
```

> When entering the input via the keyboard, `ctrl+d` will end the stream.

You can also provide more than one input file name:

Copy

```
jadi@funlife:~/w/lpic/101$ cat mydata directory_data
test
this is the second line
bye
total 0
-rw-rw-r-- 1 jadi jadi 0 Jan  4 17:33 12
-rw-rw-r-- 1 jadi jadi 0 Jan  4 17:33 62
-rw-rw-r-- 1 jadi jadi 0 Jan  4 17:33 neda
-rw-rw-r-- 1 jadi jadi 0 Jan  4 17:33 jadi
-rw-rw-r-- 1 jadi jadi 0 Jan  4 17:33 you
-rw-rw-r-- 1 jadi jadi 0 Jan  4 17:34 amir
-rw-rw-r-- 1 jadi jadi 0 Jan  4 17:37 directory_data
```

Some common cat switches are `-n` to show line numbers, `-s` to squeeze blanks, `-T` to show tabs, and `-v` to show non-printing characters.

bzcat, xzcat, zcat, gzcat

There are used to directly `cat` the bz, xz, and Z & gz compressed files. These let you see the contents of compressed files without uncompressing them first.

less

This is a powerful tool to view larger text files. It can paginate, search and move in text files.

> There is another command called `more`. It's more familiar for people coming from the DOS environment and not very common in the Linux world. Do not use it. Remember: `less` is more than `more`.

Some less common commands are as follows.

Command	Usage
q	Exit
/foo	Searches for foo
n	Next (search)
N	Previous (search)
?foo	Search backward for foo
G	Go to end
nG	Go to line n
PageUp, PageDown, UpArrow, DownArrow	You guess!

od

This command _dump_s files (Shows files in formats other than text). Normal behavior is OctalDump (showing in base 8):

Copy

```
jadi@funlife:~/w/lpic/101$ od mydata
0000000 062564 072163 072012 064550 020163 071551 072040 062550
0000020 071440 061545 067543 062156 066040 067151 005145 074542
0000040 005145
0000042
```

Not good enough for normal human beings. Let's use some switches:

- **-t** will tell what format to print:
  `-t a` for showing only named characters
  `-t c` for showing escaped chars.
  You can summarize the two above to `-a` and `-c`
- **-A** for choosing how to present offset field:
  `-A d` for Decimal,
  `-A o` for Octal,
  `-A x` for hex
  `-A n` for None

  `od` is very useful to find problems in your text files - Say finding out if you are using tabs or correct line endings.

Choosing parts of files

split

Will split the files. It is very useful for transferring HUGE files on smaller media (say splitting a 3TB file into 8GB parts and moving them to another machine with a USB Disk).

Copy

```
jadi@funlife:~/w/lpic/101$ cat mydata
hello
this is the second line
but as you can see we are
still writing
and this is getting longer
.
.
and longer
and longer!
jadi@funlife:~/w/lpic/101$ ls
mydata
jadi@funlife:~/w/lpic/101$ split -l 2 mydata
jadi@funlife:~/w/lpic/101$ ls
mydata    xaa  xab  xac  xad  xae
jadi@funlife:~/w/lpic/101$ cat xab
but as you can see we are
still writing
```

- By default, split uses xaa, xab, xac, ... for output file names. It can be changed with `split -l 2 mydata output` which split mydata into outputaa, outputab, ...; 2 lines per file.
- the `-l 2` splits 2 lines per file. It is possible to use `-b 42` to split every 42 bytes or even `-n 5` to force 5 output files.
- If you want numeric output (x00, x01, ..) use `-d` option.

  Need to join these files? `cat` them with `cat x* > originalfile`.

head and tail

Will show the beginning (head) or end (tail) of text files. By default, it will show 10 lines but you can change it by `-n20` or `-20`.

> `tail -f` follows the new lines which are being written at the end of the file. Very useful.

cut

The `cut` command will *cut* one or more columns from a file. Good for separating fields:

Lets cut the *first field* of a file.

Copy

```
jadi@funlife:~/w/lpic/101$ cat howcool
jadi    5
sina    6
rubic    2
you     12
jadi@funlife:~/w/lpic/101$ cut -f1 howcool
jadi
sina
```

```
rubic
you
```

Default delimiter is TAB. use `-dx` to change it to "x" or `-d' '` to change it to space

It is also possible to *cut* fields 1, 2, and 3 with `-f1-3` or only characters with index 4, 5, 7, 8 from each line `-c4,5,7,8`.

Modifying streams

nl

This command is for showing line numbers.

Copy

```
jadi@funlife:~/w/lpic/101$ nl mydata  | head -3
     1    hello
     2    this is the second line
     3    but as you can see we are
```

`cat -n` will also number lines.

sort & uniq

Will sorts its input(s).

Copy

```
jadi@funlife:~/w/lpic/101$ cat uses
you fedora
jadi ubuntu
rubic windows
neda mac
jadi@funlife:~/w/lpic/101$ cat howcool
jadi    5
sina    6
rubic    2
you     12
jadi@funlife:~/w/lpic/101$ sort howcool uses
jadi    5
jadi ubuntu
neda mac
rubic    2
rubic windows
sina    6
you     12
```

If you want a reverse sort, use the `-r` switch.

If you want to sort NUMERICALLY (so 9 is lower than 19), use `-n`.

And the `uniq` removes duplicate entries from its input. Normal behavior is removing only the duplicated lines but you can change its behavior, for example `-f1` switch forces it not to check the first field.

Copy

```
jadi@funlife:~/w/lpic/101$ uniq what_i_have.txt
laptop
socks
tshirt
ball
socks
glasses
jadi@funlife:~/w/lpic/101$ sort what_i_have.txt | uniq
ball
glasses
laptop
socks
tshirt
jadi@funlife:~/w/lpic/101$
```

As you can see, the input HAVE TO BE sorted for uniq to work.

uniq has great switches:

Copy

```
jadi@funlife:~/w/lpic/101$ cat what_i_have.txt
laptop
socks
tshirt
ball
socks
glasses
jadi@funlife:~/w/lpic/101$ sort what_i_have.txt  | uniq -c  #show count of each item
      1 ball
      1 glasses
      1 laptop
      2 socks
      1 tshirt
jadi@funlife:~/w/lpic/101$ sort what_i_have.txt  | uniq -u #show only non-repeated items
ball
glasses
laptop
tshirt
jadi@funlife:~/w/lpic/101$ sort what_i_have.txt  | uniq -d #show only repeated items
socks
```

paste

The paste command pastes lines from two or more files side-by-side! You cannot do this in a general text editor with ease!

Copy

```
jadi@funlife:~/w/lpic/101$ cat howcool
jadi    5
sina    6
rubic    2
you     12
jadi@funlife:~/w/lpic/101$ cat uses
you fedora
jadi ubuntu
rubic windows
neda mac
jadi@funlife:~/w/lpic/101$ paste howcool uses
jadi    5    you fedora
sina    6    jadi ubuntu
rubic    2    rubic windows
you     12    neda mac
```

tr

The `tr` command *translates* characters in the stream. For example, `tr 'ABC' '123'` will replace A with 1, B with 2, and C with 3 in the provided stream. It is a pure filter and does not accept the input file name. If needed you can pipe the cat with it (see chapter 103.4).

Copy

```
jadi@funlife:~/w/lpic/101$ cat mydata
hello
this is the second line
but as you can see we are
still writing
and this is getting longer
.
.
```

```
and longer
and longer!
jadi@funlife:~/w/lpic/101$ cat mydata | tr 'and' 'AND'
hello
this is the second liNe
but As you cAN see we Are
still writiNg
AND this is gettiNg loNger
.
.
AND loNger
AND loNger!
```

Note: all **'a'**s are replaced with **'A'**.

sed

sed is **s**tream **ed**itor. It is POWERFUL and can do things that are not far from magic! Just like most of the tools we've seen far now, sed can work as a filter or take input from a file. Sed is a great tool for replacing text with using **regular expressions** . If you need to replace A with B only once in each line in a stream just issue `sed 's/A/B/'`:

Copy

```
jadi@funlife:~/w/lpic/101$ cat uses
you fedora
jadi ubuntu
rubic windows
neda mac
jadi@funlife:~/w/lpic/101$ sed 's/ubuntu/debian/' uses
you fedora
jadi debian
rubic windows
neda mac
jadi@funlife:~/w/lpic/101$
```

The pattern for changing EVERY occurrence of A to B in a line is `sed 's/A/B/g'`.

Remember escape characters? They also work here and this will remove every *new line* from a file and will replace it with a space:

Copy

```
jadi@funlife:~/w/lpic/101$ cat mydata
hello
this is the second line
but as you can see we are
still writing
and this is getting longer
.
.
and longer
and longer!
jadi@funlife:~/w/lpic/101$ sed 's/ /\t/g' mydata > mydata.tab
jadi@funlife:~/w/lpic/101$ cat mydata.tab
hello
this    is the second    line
but    as    you    can    see    we    are
still    writing
and    this    is    getting    longer
.
.
and    longer
and    longer!
```

Getting stats

wc

The `wc` is *word count*. It counts the lines, words, and bytes in the input stream.

Copy

```
jadi@funlife:~/w/lpic/101$ wc mydata
  9  25 121 mydata
```

> It is very common to count the line numbers with `-l` switch.

-

You should know that if you put `-` instead of a filename, the data will be replaced from the pipe (or keyboard stdin).

Copy

```
jadi@funlife:~/w/lpic/101$ wc -l mydata | cat mydata - mydata
hello
this is the second line
but as you can see we are
still writing
and this is getting longer
.
.
and longer
and longer!
9 mydata
hello
this is second line
but as you can see we are
still writing
and this is getting longer
.
.
and longer
and longer!
```

Hashing

A hash function is any function that can be used to map data of arbitrary size to fixed-size values. There are different hashes and we use them for different purposes. For example, a site may hash your password in its database to keep it secure (and check the hash of provided password with a hash it already has in DB during logins) a site may provide the hash of a file so you can be sure that you've downloaded the correct file and ...

The hashing algorithms covered in LPIC1 are:

- md5sum
- sha256sum
- sha512sum

You can check any file (or input streams hash with something like this):

Copy

```
jadi@ocean:~$ md5sum /tmp/myfile.txt
8183aa57a23658efe7ba7aebe60816bc  /tmp/myfile.txt
```

```
jadi@ocean:~$ sha256sum /tmp/myfile.txt
7ddcfda184b55ee06b0c81e0ad136b1aa4a86daeb1078bcaeccc246eb2c8693b  /tmp/myfile.txt
jadi@ocean:~$ sha512sum /tmp/myfile.txt
79e5d789528e5e55fc1bddcb381afd56e896b1b452347a76777fb38d76c9754278700036f35df2a53c4d53d3e3623538a8b9ed155a3fd5275e667bdbf3c0b359  /tmp/myfile
```

As you can see, `sha512sum` creates a longer hash which is more secure.

103.3 Perform basic file management

Weight: 4

Candidates should be able to use the basic Linux commands to manage files and directories.

Objectives

- Copy, move and remove files and directories individually.
- Copy multiple files and directories recursively.
- Remove files and directories recursively.
- Use simple and advanced wildcard specifications in commands.
- Using find to locate and act on files based on type, size, or time.
- Usage of tar, cpio, and dd.

Terms

- cp
- find
- mkdir
- mv
- ls
- rm
- rmdir
- touch
- tar
- cpio
- dd
- file
- gzip
- gunzip
- bzip2
- bunzip2
- xz
- unxz
- file globbing

Wildcards and file globbing

File globbing is a shell capability that lets you tell things like : - All files - everything which starts with A - all files with 3-letter names which end in A or B or C - ...

To do so you need to know about these characters:

- `*` means **any string**
- `?` means any single character
- `[ABC]` matches A, B, or C
- `[a-k]` matches a, b, c, ..., k (both lower-case and upper-case)
- `[0-9a-z]` matches all digits and numbers
- `[!x]` means NOT X.

knowing these, you can create your patterns. For example:

command	meaning
rm *	delete all files in this directory
ls A*B	show all files starting with A and ending with B
cp ???.* /tmp	Copy all files with 3 characters, then a dot then whatever (even nothing) to /tmp
rmdir [a-z]*	remove all empty directories which start with a letter

general commands

listing with `ls`

`ls` used to *list* directories & files. You can provide an absolute or relative path; if omitted the "." will be used as a target.

Copy

```
jadi@lpicjadi:~/lpic1-practice-iso/100/103.3$ ls -ltrh
total 16K
-rw-rw-r-- 1 jadi jadi 207 Aug 14 04:43 tasks.txt
-rw-rw-r-- 1 jadi jadi  29 Aug 14 04:43 info.txt
-rw-rw-r-- 1 jadi jadi  24 Aug 14 04:44 data.txt
-rw-rw-r-- 1 jadi jadi 116 Aug 14 04:44 note_to_self
```

> First field indicates if this is a file (`-`) or directory (`d`).

Some common switches are:

- `-l` is for *long* (more info for each file)
- `-1` will print one file per line
- `-t` sorts based on modification date
- `-r` reverses the search (so `-tr` is reverse time (newer files at the bottom).

> you can mix switches. A famous one is `-ltrh` (long + human readable sizes + reverse time).

Copy (`cp`), Move (`mv`), and Delete (`rm`)

cp

This will *copy* files from one place/name to another place/name. If the target is a directory, all sources will be copied there.

Copy

```
cp source destination
```

A common switch is `-r` (or `-R`) which copies recursively (directories and their contents). So for copying a directory called `A` to `/tmp/` you can issue `cp -r A /tmp/`.

mv

Will *move* or *rename* files or directories. It works like `cp` command. If you are moving a file on the same file system, the **inode** won't change.

In general:

- If the target is an existing directory, then all sources are copied into the target
- If the target directory does not exist, then the source must be only one directory which will be renamed to the target directory.
- If the target is a file, then the source must be only one file so rename will happen.

These look like "formulas" but they are common sense!

rm

Removes (Deletes) **files**. You can do this recursively using the `-r` switch or even prevent it from checking for confirmations using `-f` (force) switch. So a `rm -rf /` means *delete everything from the file system*.

notes

Normally, the cp command will copy a file over an existing copy, if the existing file is writable. On the other hand, the `mv` will not move or rename a file if the target exists. Although this is highly dependent on your system's configuration. But in all cases, you can overcome this using the `-f` switch.

- `-f` (--force) will cause cp to try overwriting the target.
- `-i` (--interactive) will ask Y/N question (deleting / overwriting).
- `-b` (--backup) will make backups of overwritten files
- `-p` will *preserve* the attributes.

Creating (mkdir) and removing (rmdir) directories

The `mkdir` command creates directories.

Copy

```
jadi@lpicjadi:~/lpic1-practice-iso/100/103.3$ ls -ltrh
total 16K
-rw-rw-r-- 1 jadi jadi 207 Aug 14 04:43 tasks.txt
-rw-rw-r-- 1 jadi jadi  29 Aug 14 04:43 info.txt
-rw-rw-r-- 1 jadi jadi  24 Aug 14 04:44 data.txt
-rw-rw-r-- 1 jadi jadi 116 Aug 14 04:44 note_to_self
jadi@lpicjadi:~/lpic1-practice-iso/100/103.3$ mkdir new_dir
jadi@lpicjadi:~/lpic1-practice-iso/100/103.3$ ls -ltrh
total 20K
-rw-rw-r-- 1 jadi jadi  207 Aug 14 04:43 tasks.txt
-rw-rw-r-- 1 jadi jadi   29 Aug 14 04:43 info.txt
-rw-rw-r-- 1 jadi jadi   24 Aug 14 04:44 data.txt
-rw-rw-r-- 1 jadi jadi  116 Aug 14 04:44 note_to_self
drwxrwxr-x 2 jadi jadi 4.0K Aug 14 04:57 new_dir
```

If you want to create a tree of directories, you can use `-p` switch to tell the `mkdir` to create the *parent* directories if needed:

Copy

```
jadi@lpicjadi:~/lpic1-practice-iso/100/103.3$ ls -ltrh
total 20K
-rw-rw-r-- 1 jadi jadi  207 Aug 14 04:43 tasks.txt
-rw-rw-r-- 1 jadi jadi   29 Aug 14 04:43 info.txt
-rw-rw-r-- 1 jadi jadi   24 Aug 14 04:44 data.txt
-rw-rw-r-- 1 jadi jadi  116 Aug 14 04:44 note_to_self
drwxrwxr-x 2 jadi jadi 4.0K Aug 14 04:57 new_dir
jadi@lpicjadi:~/lpic1-practice-iso/100/103.3$ mkdir -p 1/2/3
jadi@lpicjadi:~/lpic1-practice-iso/100/103.3$ tree
.
├── 1
│   └── 2
│       └── 3
├── data.txt
├── info.txt
├── new_dir
├── note_to_self
└── tasks.txt
```

If you need to delete a directory the command is `rmdir` and you can also use the -p for nested removing:

Copy

```
jadi@lpicjadi:~/lpic1-practice-iso/100/103.3$ rmdir -p 1/2/3
jadi@lpicjadi:~/lpic1-practice-iso/100/103.3$ rmdir new_dir
jadi@lpicjadi:~/lpic1-practice-iso/100/103.3$ tree
.
├── data.txt
├── info.txt
├── note_to_self
└── tasks.txt

0 directories, 4 files
```

> If you are using `rmdir` to remove a directory, it *MUST BE EMPTY*! That's why many people use `rm -rf directory_name` to delete the not-empty directory and whatever is in it.

touch

The `touch` will create an empty file (if it does not exists) or updates the **modification** date of a file if it already exists. The default time is *now* but you can specify other times too.

Copy

```
jadi@lpicjadi:~/lpic1-practice-iso/100/103.3$ ls -ltrh
total 16K
-rw-rw-r-- 1 jadi jadi 207 Aug 14 04:43 tasks.txt
-rw-rw-r-- 1 jadi jadi  29 Aug 14 04:43 info.txt
-rw-rw-r-- 1 jadi jadi  24 Aug 14 04:44 data.txt
-rw-rw-r-- 1 jadi jadi 116 Aug 14 04:44 note_to_self
jadi@lpicjadi:~/lpic1-practice-iso/100/103.3$ touch new_file
jadi@lpicjadi:~/lpic1-practice-iso/100/103.3$ ls -ltrh
total 16K
-rw-rw-r-- 1 jadi jadi 207 Aug 14 04:43 tasks.txt
-rw-rw-r-- 1 jadi jadi  29 Aug 14 04:43 info.txt
-rw-rw-r-- 1 jadi jadi  24 Aug 14 04:44 data.txt
-rw-rw-r-- 1 jadi jadi 116 Aug 14 04:44 note_to_self
-rw-rw-r-- 1 jadi jadi   0 Aug 14 05:08 new_file
jadi@lpicjadi:~/lpic1-practice-iso/100/103.3$ touch note_to_self
jadi@lpicjadi:~/lpic1-practice-iso/100/103.3$ ls -ltrh
total 16K
-rw-rw-r-- 1 jadi jadi 207 Aug 14 04:43 tasks.txt
-rw-rw-r-- 1 jadi jadi  29 Aug 14 04:43 info.txt
-rw-rw-r-- 1 jadi jadi  24 Aug 14 04:44 data.txt
-rw-rw-r-- 1 jadi jadi   0 Aug 14 05:08 new_file
-rw-rw-r-- 1 jadi jadi 116 Aug 14 05:08 note_to_self
```

Or you can specify times. It is possible to use `-d` and give dates or use `-t` and give a timestamp in the form of `[[CC]YY]MMDDhhmm[.ss]`

Copy

```
$ touch -t 200908121510.59 file1
$ touch -d 11am file2
$ touch -d "last fortnight" file3
$ touch -d "yesterday 6am" file4
$ touch -d "2 days ago 12:00" file5
$ touch -d "tomorrow 02:00" file6
$ touch -d "5 Nov" file3
$ ls -ltrh file?
-rw-rw-r-- 1 jadi jadi 0 Aug 12  2009 file1
-rw-rw-r-- 1 jadi jadi 0 Aug 12 12:00 file5
-rw-rw-r-- 1 jadi jadi 0 Aug 13 06:00 file4
-rw-rw-r-- 1 jadi jadi 0 Aug 14  2022 file2
-rw-rw-r-- 1 jadi jadi 0 Aug 15  2022 file6
-rw-rw-r-- 1 jadi jadi 0 Nov  5  2022 file3
```

Oh.. and it's possible to use another file's time to set, with switch `-r` (for --reference)):

Copy

```
jadi@lpicjadi:~/lpic1-practice-iso/100/103.3$ ls -l /etc/debian_version
-rw-r--r-- 1 root root 13 Aug 22  2021 /etc/debian_version
jadi@lpicjadi:~/lpic1-practice-iso/100/103.3$ touch -r /etc/debian_version file1
jadi@lpicjadi:~/lpic1-practice-iso/100/103.3$ ls -ltrh
total 20K
-rw-rw-r-- 1 jadi jadi   0 Aug 22  2021 file1
```

file

To determine the type of a file, you should use `file` command. It looks *into* the file and determines its type.

Copy

```
jadi@lpicjadi:~/lpic1-practice-iso/100/103.3$ file file1
file1: empty
jadi@lpicjadi:~/lpic1-practice-iso/100/103.3$ file note_to_self
note_to_self: ASCII text
jadi@lpicjadi:~/lpic1-practice-iso/100/103.3$ file /bin/bash
/bin/bash: ELF 64-bit LSB pie executable, x86-64, version 1 (SYSV), dynamically linked, interpreter /lib64/ld-linux-x86-64.so.2, BuildID[sha1
```

> `-i` switch prints the mime format

dd

The `dd` command copies data from its input to its output (say files or devices). You may use it just like copy:

Copy

```
jadi@lpicjadi:~/lpic1-practice-iso/100/103.3$ dd if=note_to_self of=new_file
0+1 records in
0+1 records out
116 bytes copied, 0.00141561 s, 81.9 kB/s
jadi@lpicjadi:~/lpic1-practice-iso/100/103.3$ cat new_file
I will continue learning... and if I get confused, I'll repeat the last section once more till everything is clear!
jadi@lpicjadi:~/lpic1-practice-iso/100/103.3$
```

- `if` is the Input File
- `of` is the Output File

But commonly people use it to read/write from block devices. For example, this will read all the sectors from the `/dev/sdb` and will write them to a file named`backup.dd`. Later you can restore this backup by swapping the `if` and `of` and writing from the `backup.dd` to `/dev/sdb`.

Copy

```
# dd if=/dev/sda of=backup.dd bs=4096
```

or even:

Copy

```
# dd if=/dev/sda2 | gzip > backup.dd.gzip
```

Another common usage is creating files of specific sizes:

Copy

```
$ dd if=/dev/zero of=1g.bin bs=1G count=1
```

or even *writing* your iso files to a USB disk to have a live bootable USB:

Copy

```
$ sudo dd if=ubuntu.iso of=/dev/sdc bs=2048
```

> Caution: here you are writing directly on a block device. If you do something wrong... you will ruin your disk and need to reformat it.

find

The `find` command helps us to find files based on different criteria. Look at this:

Copy

```
$ find . -iname "[a-j]*"
./howcool.sort
./alldata
./mydir/howcool.sort
./mydir/newDir/insideNew
./howcool
```

- The first parameter says where should we search (including subdirectories).
- The `-name` switch indicates the criteria (here `iname` means search files with this name and ignore the character cases (z equals Z)).

Another common switch is `-type` to indicate the type we are searching for (`f` for regular files, `d` for directories, and `l` for symbolic links):

Copy

```
jadi@lpicjadi:~/lpic1-practice-iso/100/103.3$ find . -type d -iname "[a-j]*"
./directory
./directory/innder_one
```

if you want to search for file sizes do as below:

command	meaning
-size 100c	files which are exactly 100 characters/bytes (you can also use `b`)
-size +100k	files which are more than 100 kilobytes
-size -20M	files smaller than 20Megabytes
-size +2G	files bigger than 2Gigabytes

So this will find all files ending in *tmp* with sizes between 1M and 100M in /var/ directory:

Copy

```
find /var -iname '*tmp' -size +1M -size -100M
```

you can find all empty files with `find . -size 0b` or `find . -empty`

Another useful search criterion is time. These are some of the options:

switch	meaning	samples
-amin	Access Minutes	`-amin 40` means "files accessed exactly 40min ago" or `-amin +40` mins files accessed more than 40min ago and `-amin -40` means files accessed less than 40min ago
-cmin	Status Change Min	`-cmin +60` file status changed before last hour
-mmin	Modified Minutes	`-mmin -60` will give us files modified in last hour
-atime	access time in days	`-atime +1` means files access "more than 1 days ago (which means 2 days and more)
-ctime	Status Changed in Days	
-mtime	Modified days	
-newer	Newer than reference	`-newer file1` will give you files which are newer than file1

if you add `-daystart` switch to -mtime or -atime it means that we want to consider days as calendar days, starting at midnight.

Acting on files

We can execute commands or do other actions on files with various switches:

switch	meaning
-ls	will run ls -dils on each file
-print	will print the full name of the files on each line

But the best way to run commands on found files is `-exec` switch. You can point to the file with `'{}'` or `{}` and finish your command with `\;`.

For example, This will remove all empty files in this directory and its subdirectories:

Copy

```
find . -empty -exec rm '{}' \;
```

or this will rename all htm files to html

Copy

```
find . -name "*.htm" -exec mv '{}' '{}l' \;
```

since deleting found files is a common task, there is a switch for it: `-delete`

Compression

gzip & gunzip

Straight forward, one gzips file and one ungzips file; In place:

Copy

```
jadi@lpicjadi:~/lpic1-practice-iso/100/103.3$ ls -ltrh
total 20K
-rw-rw-r-- 1 jadi jadi  207 Aug 14 04:43 tasks.txt
-rw-rw-r-- 1 jadi jadi   29 Aug 14 04:43 info.txt
-rw-rw-r-- 1 jadi jadi   24 Aug 14 04:44 data.txt
-rw-rw-r-- 1 jadi jadi    0 Aug 14 05:08 new_file
-rw-rw-r-- 1 jadi jadi  116 Aug 14 05:08 note_to_self
drwxrwxr-x 3 jadi jadi 4.0K Aug 14 05:20 directory
jadi@lpicjadi:~/lpic1-practice-iso/100/103.3$ ls -ltrh
total 20K
-rw-rw-r-- 1 jadi jadi  171 Aug 14 04:43 tasks.txt.gz
-rw-rw-r-- 1 jadi jadi   29 Aug 14 04:43 info.txt
-rw-rw-r-- 1 jadi jadi   24 Aug 14 04:44 data.txt
-rw-rw-r-- 1 jadi jadi    0 Aug 14 05:08 new_file
-rw-rw-r-- 1 jadi jadi  116 Aug 14 05:08 note_to_self
drwxrwxr-x 3 jadi jadi 4.0K Aug 14 05:20 directory
jadi@lpicjadi:~/lpic1-practice-iso/100/103.3$ gunzip tasks.txt.gz
jadi@lpicjadi:~/lpic1-practice-iso/100/103.3$ ls -ltrh
total 20K
-rw-rw-r-- 1 jadi jadi  207 Aug 14 04:43 tasks.txt
-rw-rw-r-- 1 jadi jadi   29 Aug 14 04:43 info.txt
-rw-rw-r-- 1 jadi jadi   24 Aug 14 04:44 data.txt
-rw-rw-r-- 1 jadi jadi    0 Aug 14 05:08 new_file
-rw-rw-r-- 1 jadi jadi  116 Aug 14 05:08 note_to_self
drwxrwxr-x 3 jadi jadi 4.0K Aug 14 05:20 directory
```

- gzip preserves time
- gzip creates the new compressed file with the same name but with .gz ending
- gzip removes the original files after creating the compressed file (you can keep the input file with `-k` switch)

bzip2 & bunzip2

The `bzip2` is another compressing tool. Works just like the famous `gzip` but with a different compression algorithm.

Copy

```
jadi@lpicjadi:~/lpic1-practice-iso/100/103.3$ bzip2 tasks.txt
jadi@lpicjadi:~/lpic1-practice-iso/100/103.3$ ls  -ltrh
total 20K
-rw-rw-r-- 1 jadi jadi  172 Aug 14 04:43 tasks.txt.bz2
-rw-rw-r-- 1 jadi jadi   29 Aug 14 04:43 info.txt
-rw-rw-r-- 1 jadi jadi   24 Aug 14 04:44 data.txt
-rw-rw-r-- 1 jadi jadi    0 Aug 14 05:08 new_file
-rw-rw-r-- 1 jadi jadi  116 Aug 14 05:08 note_to_self
drwxrwxr-x 3 jadi jadi 4.0K Aug 14 05:20 directory
jadi@lpicjadi:~/lpic1-practice-iso/100/103.3$ bunzip2 tasks.txt.bz2
jadi@lpicjadi:~/lpic1-practice-iso/100/103.3$ ls
data.txt  directory  info.txt  new_file  note_to_self  tasks.txt
```

xz & unxz

Another compression/decompression tool is just like `gzip` and `bzip2`.

Copy

```
jadi@lpicjadi:~/lpic1-practice-iso/100/103.3$ xz tasks.txt
jadi@lpicjadi:~/lpic1-practice-iso/100/103.3$ ls -ltrh
total 24K
-rw-rw-r-- 1 jadi jadi  224 Aug 14 04:43 tasks.txt.xz
-rw-rw-r-- 1 jadi jadi   29 Aug 14 04:43 info.txt
-rw-rw-r-- 1 jadi jadi   24 Aug 14 04:44 data.txt
-rw-rw-r-- 1 jadi jadi  116 Aug 14 05:08 note_to_self
drwxrwxr-x 3 jadi jadi 4.0K Aug 14 05:20 directory
-rw-rw-r-- 1 jadi jadi  116 Aug 14 07:51 new_file
jadi@lpicjadi:~/lpic1-practice-iso/100/103.3$ unxz tasks.txt.xz
jadi@lpicjadi:~/lpic1-practice-iso/100/103.3$ ls -ltrh
total 24K
-rw-rw-r-- 1 jadi jadi  207 Aug 14 04:43 tasks.txt
-rw-rw-r-- 1 jadi jadi   29 Aug 14 04:43 info.txt
-rw-rw-r-- 1 jadi jadi   24 Aug 14 04:44 data.txt
-rw-rw-r-- 1 jadi jadi  116 Aug 14 05:08 note_to_self
drwxrwxr-x 3 jadi jadi 4.0K Aug 14 05:20 directory
-rw-rw-r-- 1 jadi jadi  116 Aug 14 07:51 new_file
```

Please note that *compressing* a small text file makes it larger. This is *normal* in small files because of all the headers and metadata.

> In some cases, commands like `unxz` is just a calls to `xz --dcompress`

Archiving with tar & cpio

Sometimes we need to create an archive file container of many other files. This operation is different than compressing, it combines files into one and then extracts them again. Archiving is mostly used in backups, moving files to a new location (say via email), and such. This is done with `cpio` and `tar`.

tar

TapeARchive or tar is the most common archiving tool. It automatically creates an archive file from a directory and all its subdirs.

Common switches are:

switch	meanint
-cf `myarchive.tar`	create file named myarchive.tar
-xf `myarchive.tar`	extract a file named myarchive.tar
-z	compress the archive with gzip after creating it
-j	compress the archive with bzip2 after creating it
-v	verbose! print a lot of data about what is happening
-r	append new files to the currently available archive

> If you issue absolute paths, tar removes the starting slash (/) for safety reasons when creating an archive. If you want to override, use -p option.
>
> tar can work with tapes and other storages. That's why we use `-f` to tell it that we are working with files.

cpio

Gets a list of files and creates an archive (one file). This file can be used later to extract the original files.

Copy

```
$ ls | cpio -o > allfilesls.cpio
3090354 blocks
```

- `-o` tells cpio to create an output from its input

Please note that `cpio` does not look into the folders. So mostly we use it with find:

Copy

```
find . -name "*" | cpio -o > myarchivefind.cpio
```

To extract the original files:

Copy

```
mkdir extract
mv myarchivefind.cpio extract
cd extract
cpio -id < myarchivefind.cpio
```

- `-d` will create the folders
- `-i` is for extract

103.4 Use streams, pipes, and redirects

Weight: 4

Candidates should be able to redirect streams and connect them to efficiently process textual data. Tasks include redirecting standard input, standard output, and standard error, piping the output of one command to the input of another command, using the output of one command as arguments to another command, and sending output to both stdout and a file.

Objectives

- Redirecting standard input, standard output, and standard error.
- Pipe the output of one command to the input of another command.
- Use the output of one command as arguments to another command.
- Send output to both stdout and a file.

Terms

- tee
- xargs

These features help us to control the input/output of the commands and do things like saving the output of a command to a file, getting the input of a command from another command, or separating the normal output from errors. We've already used them in previous sections but let's learn more and deepen our understanding of these.

Redirecting standard IO

On a Linux system, most shells use streams for input and output. These streams can be from (and toward) various things including keyboards, block devices (hards, USB sticks, ..), files, and ...

We have 3 different standard streams:

0. **STDIN** is the standard input stream, which provides input to a command.
1. **STDOUT** is the standard output stream, which includes the output of a command.
2. **STDERR** is the standard error stream, which includes the error output of a command.

   The `0`, `1` & `2` numbering, indicates the ***STDIN***, ***STDOUT*** and ***STDERR** accordingly. For example, if you want to redirect the stderror, you can use 2> and the STDERR will be redirected.

These are the other redirections you can use:

Operator	Usage
>	Redirect STDOUT to a file; Overwrite if exists
>>	Redirect STDOUT to a file; Append if exists
2>	Redirect STDERR to a file; Overwrite if exists
2>>	Redirect STDERR to a file; Append if exists
&>	Redirect both STDOUT and STDERR; Overwrite if exists
&>>	Redirect both STDOUT and STDERR; Append if exists
<	Redirect STDIN from a file
<>	Redirect STDIN from the file and send the STDOUT to it

Some examples:

Copy

```
$ ls
bob jack    jadi    linus   sara who_uses_what.txt
$ ls x*
ls: x*: No such file or directory
$ ls j*
jack    jadi
$ ls j* x* > output 2> errors
$ cat output
jack
jadi
$ cat errors
ls: x*: No such file or directory
$ cat who_uses_what.txt
jadi, fedora
linux, fedora
bob, ubuntu
jack, arch
sara, fedora
$ tr ' ', '' < who_uses_what.txt
tr: empty string2
$ cat who
$ tr ',', '|' < who_uses_what.txt
jadi| fedora
linux| fedora
bob| ubuntu
jack| arch
sara| fedora
```

It is also possible to use `&1` and `&2` and `&0` to refer to the **target** of STDOUT, STDERR & STDIN. In this case `ls > file1 2>&1` means *redirect output to file1 and output stderr to same place as stdout (file1)*

> Be careful! `ls 2>&1 > file1` means *print stderr to the current location of stdout (terminal) and then change the stdout to file1*

sending to null

In Linux **/dev/null** device works like an abyss. You can send anything there and it disappears without being any burden on your system. So it is normal to say:

Copy

```
$ ls j* x* > file1
ls: x*: No such file or directory
$ ls j* x* > file1 2>/dev/null
$ cat file1
jack
jadi
```

here-documents

Many shells have here-documents (also called here-docs) as a way of input. You use `<<` and a `WORD` and then whatever you input is considered stdin till you give only the WORD in one line.

Copy

```
$ tr ' ' '.' << END_OF_DATA
> this is a line
> and then this
```

```
>
> we'll still type
> and,
> done!
> END_OF_DATA
this.is.a.line
and.then.this

we'll.still.type
and,
done!
```

Here-Documents are very useful if you are writing scripts and automated tasks.

Pipes

With the pipe (`|`), you can redirect STDOUT, STDIN, and STDERR between multiple commands all on one command line. When you do `command1 | command2`; command1, is executed but its STDOUT is redirected as STDIN into the COMMAND2.

Copy

```
$ cat who_uses_what.txt
jadi, fedora
linux, fedora
bob, ubuntu
jack, arch
sara, fedora
$ cut -f2 -d, who_uses_what.txt | sed -e 's/ //g' | sort  | uniq -c | sort -nr
   3 fedora
   1 ubuntu
   1 arch
```

If you need to start your pipeline with the contents of a file, start with `cat filename | ...` or use a `<` stdin redirect.

Pipes are one of the super strong & super amazing features in the UNIX world. They let you create *new* tools by combining tools that do atomic things. As an example, check this out:

xargs

The xargs utility reads space, tab, newline and end-of-file delimited strings from the standard input and executes the provided utility with the strings as their arguments.

Copy

```
$ ls
bob         file1           jadi            output          who_uses_what.txt
errors          jack            linus           sara
$ ls | xargs echo these are files:
these are files: bob errors file1 jack jadi linus output sara who_uses_what.txt
```

> if you do not give any command to the `xargs` , the echo will be the default command.

One common switch is `-I`. This is useful if you need to pass stdin arguments in the middle (or even start) of your commands. Use it like this: `xargs -I SOMETHING echo here is SOMETHING end`:

Copy

```
$ cat who_uses_what.txt
jadi, fedora
linus, fedora
bob, ubuntu
jack, arch
sara, fedora
$ cat who_uses_what.txt | xargs -I DATA echo name is DATA is the choice.
name is jadi, fedora is the choice.
name is linus, fedora is the choice.
name is bob, ubuntu is the choice.
name is jack, arch is the choice.
name is sara, fedora is the choice.
```

Two more useful switches: 1. `-L 1` breaks based on new lines 2. `-n 1` tells xargs to invoke the provided utility after receiving 1 argument.

tee

The problem with redirection is that you cannot see the progress of your commands in the same terminal. The `tee` utility solves this. If you need to see the output on screen and also save it to a file, `tee` is your friend. Give it one or more filenames and it will do the trick.

Copy

```
$ ls -1 | tee allfiles myfiles
bob
errors
file1
jack
jadi
linus
output
sara
who_uses_what.txt
$ cat allfiles myfiles
bob
errors
file1
jack
jadi
linus
output
sara
who_uses_what.txt
bob
errors
file1
jack
jadi
linus
output
sara
who_uses_what.txt
```

The `-a` switch will append to files if they exist.

> If you need to save ***stderr*** too, first redirect it to ***stdout***

103.5 Create, monitor, and kill processes

Weight: 4

Candidates should be able to perform basic process management.

Objectives

- Run jobs in the foreground and background.
- Signal a program to continue running after logout.
- Monitor active processes.
- Select and sort processes for display.
- Send signals to processes.

Terms

- &
- bg
- fg
- jobs
- kill
- nohup
- ps
- top
- free
- uptime
- killall
- pgrep
- pkill
- watch
- screen
- tmux

Managing processes

foreground and background jobs

One of the great points of Linux even from its beginning days is the ability to run different programs and jobs at the same time. This is done by sending programs to the background.

Normally if you run a program on the terminal, it *blocks* your terminal while it's running but sending a command to the background will prevent this:

Copy

```
xeyes &
```

Even when a program is running normally in the foreground, you can do two things: - break it using `Ctrl+c` - *suspend* or pause it using `Ctrl+z`

A *stopped* job can be brought to the foreground using `fg` command (or the background using `bg`). You can also list all the jobs by the `jobs` command.

Copy

```
$ xeyes
^Z
[1]+  Stopped                 xeyes
$ jobs
[1]+  Stopped                 xeyes
$ bg
[1]+ xeyes &
$ jobs
[1]+  Running                 xeyes &
$ sleep 1000 &
[2] 7395
$ jobs
[1]-  Running                 xeyes &
[2]+  Running                 sleep 1000 &
$ fg %2
sleep 1000
^Z
[2]+  Stopped                 sleep 1000
$ jobs
[1]-  Running                 xeyes &
[2]+  Stopped                 sleep 1000
$ bg sle
[2]+ sleep 1000 &
$ jobs
[1]-  Running                 xeyes &
[2]+  Running                 sleep 1000 &
`
```

> `jobs -l` also shows the process ID of jobs

nohup

The `nohup` command lets you run your commands even after you close the terminal or logout. By default it writes its output to `nohup.out`:

Copy

```
$ nohup ping 4.2.2.4
nohup: ignoring input and appending output to ‘nohup.out’
^C$ cat nohup.out
PING 4.2.2.4 (4.2.2.4) 56(84) bytes of data.
64 bytes from 4.2.2.4: icmp_seq=1 ttl=51 time=225 ms
64 bytes from 4.2.2.4: icmp_seq=3 ttl=51 time=223 ms

--- 4.2.2.4 ping statistics ---
4 packets transmitted, 2 received, 50% packet loss, time 3010ms
rtt min/avg/max/mdev = 223.584/224.767/225.950/1.183 ms
```

> It is common to use `2>` to redirect the nohup errors to another file and use a `&` to run it in the background: `nohup script.sh > mynohup.out 2>&1 &`

kill

Despite its frightening name, the `kill` command sends unix *signals* to processes. Pressing `Ctrl+c` and `Ctrl+z` is also sending signals. By default, the `kill` command sends the signal **15** (which is TERM and tells to process to terminate itself).

Copy

```
$ jobs
[3]   Running                 xeyes &
[4]   Running                 sleep 1000 &
[5]-  Running                 sleep 2000 &
[6]+  Running                 sleep 3000 &
$ kill %4
$ jobs
[3]   Running                 xeyes &
[4]   Terminated              sleep 1000
[5]-  Running                 sleep 2000 &
[6]+  Running                 sleep 3000 &
$ jobs
[3]   Running                 xeyes &
[5]-  Running                 sleep 2000 &
[6]+  Running                 sleep 3000 &
```

It is also possible to use PIDs instead of job numbers and kill other signals. The general format is `kill -SIGNAL_ID_OR_NAME process_id`:

signal number	signal name	meaning
1	HUP	Informing the process that its controlling terminal (like an ssh connection) is terminated
15	TERM	normal termination request
9	KILL	forcefully kills the proccess

So you can do a `kill -9 8733` to force process ID 8733 to close.

> Remember the nohup command ? :) It means " do not respond to the hup signal ".

killall

This command Will send the given signal (or by default 15) to all processes with the given name:

Copy

```
$ jobs
[3]   Running                 xeyes &
[5]-  Running                 sleep 2000 &
[6]+  Running                 sleep 3000 &
$ ps -ef | grep sleep
jadi      7864  7651  0 21:07 pts/1    00:00:00 sleep 2000
jadi      7865  7651  0 21:07 pts/1    00:00:00 sleep 3000
jadi      7977  7651  0 21:14 pts/1    00:00:00 grep sleep
$ killall sleep
[5]-  Terminated              sleep 2000
[6]+  Terminated              sleep 3000
$ jobs
[3]+  Running                 xeyes &
$ ps -ef | grep sleep
jadi      7980  7651  0 21:14 pts/1    00:00:00 grep sleep
```

pkill

Will send the given signal (or 15) to all the processes with a specific pattern in their name:

Copy

```
$ jobs
[3]   Running                 xeyes &
[5]-  Running                 sleep 2000 &
[6]+  Running                 sleep 3000 &
$ ps -ef | grep sleep
jadi      7864  7651  0 21:07 pts/1    00:00:00 sleep 2000
jadi      7865  7651  0 21:07 pts/1    00:00:00 sleep 3000
jadi      7977  7651  0 21:14 pts/1    00:00:00 grep sleep
$ pkill sle
[5]-  Terminated              sleep 2000
[6]+  Terminated              sleep 3000
$ jobs
[3]+  Running                 xeyes &
$ ps -ef | grep sleep
jadi      7980  7651  0 21:14 pts/1    00:00:00 grep sleep
```

Monitoring Processes

ps

The `ps` command shows running processes on your computer. Each process has a process ID shown as **PID** and a Parent Process ID shown as PPID.

Copy

```
$ sleep 1000 &
[1] 7678
$ sleep 1001 &
[2] 7679
$ xeyes &
[3] 7680
$ ps
  PID TTY          TIME CMD
 7651 pts/1    00:00:00 bash
 7678 pts/1    00:00:00 sleep
 7679 pts/1    00:00:00 sleep
 7680 pts/1    00:00:00 xeyes
 7681 pts/1    00:00:00 ps
```

Two common switch combination is `ps aux` (or `-aux`) and `ps ef` which shows ALL processes on a system:

Copy

```
$ ps -aux | wc -l
293
```

It is also possible to use the `--sort` switch to sort the output based on different fields (+ for ascending & - for descending).

Copy

```
$ ps -af --sort +comm,-sid
UID        PID  PPID  C STIME TTY          TIME CMD
root      5486  5478  0 19:59 pts/12   00:00:00 -su
root      4444  1169  0 19:56 tty4     00:00:00 -bash
jadi      6638  5412  0 20:10 pts/0    00:00:04 node /usr/local/bin/sslocal
jadi      7778  7651  0 20:58 pts/1    00:00:00 ps -af --sort +comm,-sid
jadi      7678  7651  0 20:48 pts/1    00:00:00 sleep 1000
jadi      7679  7651  0 20:48 pts/1    00:00:00 sleep 1001
jadi      7775  7651  0 20:58 pts/1    00:00:00 sleep 1000
jadi      7776  7651  0 20:58 pts/1    00:00:00 sleep 1000
jadi      7777  7651  0 20:58 pts/1    00:00:00 sleep 1000
root      5478  5477  0 19:59 pts/12   00:00:00 su -
root      5477  5008  0 19:59 pts/12   00:00:00 sudo su -
jadi      7680  7651  0 20:48 pts/1    00:00:01 xeyes
```

pgrep

You've seen that `ps -ef` shows processes from all users. We can `grep` on that and see who is running `gedit` and what is its process ID:

Copy

```
$ ps -ef | grep gedit
jadi      6213  4604  9 20:06 ?        00:04:43 gedit
jadi      7725  7651  0 20:55 pts/1    00:00:00 grep gedit
```

but there is also a more direct way to check the PID of all gedit processes:

Copy

```
$ pgrep gedit
6213
```

top

This is the most common tool to do simple monitoring of the system. It will update the status and will give you a good glance at the status:

Copy

```
$top

top - 21:00:44 up  1:16,  5 users,  load average: 1.51, 1.65, 1.78
Tasks: 293 total,   1 running, 292 sleeping,   0 stopped,   0 zombie
%Cpu(s): 19.0 us,  5.0 sy,  0.0 ni, 70.9 id,  5.1 wa,  0.0 hi,  0.0 si,  0.0 st
KiB Mem:   8060264 total,  5359812 used,  2700452 free,   169240 buffers
KiB Swap:  7811068 total,        0 used,  7811068 free.  2250692 cached Mem

  PID USER      PR  NI    VIRT    RES    SHR S  %CPU %MEM     TIME+ COMMAND
```

```
 6570 jadi      20   0 1437752 546064  88312 S  18.4  6.8  12:00.96 firefox
 4870 jadi      20   0 1762516 299120  75664 S  12.2  3.7   7:37.05 compiz
 4492 jadi       9 -11  455152  11516   8940 S   6.1  0.1   1:06.81 pulseaudio
 4532 root      20   0  389028  77116  60192 S   6.1  1.0  12:16.63 Xorg
 4723 jadi      20   0  358936   8288   5512 S   6.1  0.1   9:51.52 ibus-daemon
 5648 jadi      20   0 1641556 203676 102840 S   6.1  2.5   3:20.88 chrome
 7082 jadi      20   0 1210748  73136  42528 S   6.1  0.9   0:36.51 Telegram
 7806 jadi      20   0   33796   3004   2500 R   6.1  0.0   0:00.02 top
    1 root      20   0   29528   4320   2584 S   0.0  0.1   0:01.71 init
```

You can see the processes, system load, uptime, CPU status, memory, ... and do some stuff:

key during top	functionality
h	help
q	quit
M	sort based on memory usage
c	show full commands
k	kill after asking pid and signal

free

The `free` command will show you info about the system memory. The default is *kilobytes* but you can change it with `-m` for megabytes, `-g` for *gigabytes* or even `-b` for bytes. You can also use the `-h` for **human readable**.

Copy

```
$ free -m
             total       used       free     shared    buffers     cached
Mem:          7871       5231       2640        332        169       2195
-/+ buffers/cache:       2866       5005
Swap:         7627          0       7627
```

> A general **hint**: If your system is using Swap, you have memory issues.

uptime

The `uptime` command shows the time, systems uptime (how long the system has been running), how many users are logged in, and the load average of 1, 5 & 15 minutes:

Copy

```
$ uptime
 21:18:52 up  1:34,  5 users,  load average: 2.38, 2.64, 2.41
```

> Although it's one of the most important KPIs of the system status, some of the experienced Linux admins do not know what the load average means. The load average shows how many processes are in the **to be run** queue. If this number is higher than the number of your CPU cores, you are in a bad situation. If it's close to the number of your cores constantly, it's kind of dangerous, and if it's less than 1/10th of your core numbers, your system is kind of idle. Do you remember how to check the number of your cores? Its in `/proc/cpuinfo` or `nproc`.

watch

Sometimes you have a command which shows you an output but you want to keep running it and observing the output. In these cases, the `watch` is your friend. It lets you run and check the output of a command in specific time intervals (default is 2 seconds).

Copy

```
$ watch free -h
```

If you have a pipe in your command, you have to quote the watched command in double-quote `"` or single-quote `'`:

Copy

```
$ watch "ls -ltrh | wc -l"
```

These are some of the switches:

- `-n` To specify the interval in seconds
- `-b` Beep if the command has a non-zero exit
- `-d` Shows the difference between runs

Terminal Multiplexers

screen

If you are used to GUI based system, it's easy to run different terminals side to side and use them to run different programs. But if you are on a server, you need other tools to multiplex your terminal. One such command is `screen`.

Run it with `screen` and press enter to exit the welcome window into a prompt. You can use it as a normal terminal and detach from it (and let it run in the background) using the Ctrl + A and then D keys. Check the list of your screens with `screen -ls` and re-attach to any of them with `screen -r screen-id`.

Below you can see a few common switches, they all should be issued after the `Ctrl + A` combination.

Key	Usage
\	Kill all processes windows and terminate the screen
|	Split current window into two vertical focuses
Shift+S	Split current window into two horizontal focuses
C	Create a window in the current focus
Tab	Go to the next focus
D	Detach from window
K	Kill current window
N	Move to Next window
P	Move to the Previous window

A great point about screen (and tmux) is the fact it remains running even after you logout of the system and its possible to relogin and re-attach to the same screen (or tmux)

tmux

Is a screen on steroids! It is not installed by default in most distributions and you have to install it first. The default command prefix is `Ctrl+B` and after running the `tmux new` you can issue these:

Key	Usage
%	Split current window vertically
"	Split current window horizontally
D	Detach from the current window
&	Kill current window

You can list the tmux sessions using `tmux ls` and re-attach to one using `tmux att` to connect to the last one or `tmux att -t session_name` to attach to a specific one.

> I highly recommend being fluent in tmux. It's super useful even when you are working locally on your machine. watch the below video for the more in-depth session:

103.6 Modify process execution priorities

Weight: 2

Candidates should be able to manage process execution priorities.

Objectives

- Know the default priority of a job that is created.
- Run a program with higher or lower priority than the default.
- Change the priority of a running process.

Terms

- nice
- ps
- renice
- top

On a Linux machine, you might have 100s of processes running at the same time and competing for more CPU & RAM. The good news is that you can give some of the processes higher or lower priority (or nice-ness) in this competition. Let's have a look at a sample `top` output:

Copy

```
$ top

Tasks: 169 total,   1 running, 168 sleeping,   0 stopped,   0 zombie
%Cpu(s):  0.0 us,  3.0 sy,  0.0 ni, 97.0 id,  0.0 wa,  0.0 hi,  0.0 si,  0.0 st
MiB Mem :   5948.9 total,   4115.2 free,    438.2 used,   1395.5 buff/cache
MiB Swap:    975.0 total,    975.0 free,      0.0 used.   5210.3 avail Mem

    PID USER      PR  NI    VIRT    RES    SHR S  %CPU  %MEM     TIME+ COMMAND
   6496 jadi      20   0   10112   3704   3212 R   6.2   0.1   0:00.01 top
      1 root      20   0  165172  10576   7780 S   0.0   0.2   0:02.80 systemd
      2 root      20   0       0      0      0 S   0.0   0.0   0:00.09 kthreadd
      3 root       0 -20       0      0      0 I   0.0   0.0   0:00.00 rcu_gp
      4 root       0 -20       0      0      0 I   0.0   0.0   0:00.00 rcu_par_gp
      6 root       0 -20       0      0      0 I   0.0   0.0   0:00.00 kworker/0:0H-events_highpri
      8 root       0 -20       0      0      0 I   0.0   0.0   0:12.71 kworker/0:1H-events_highpri
      9 root       0 -20       0      0      0 I   0.0   0.0   0:00.00 mm_percpu_wq
     10 root      20   0       0      0      0 S   0.0   0.0   0:00.00 rcu_tasks_rude_
     11 root      20   0       0      0      0 S   0.0   0.0   0:00.00 rcu_tasks_trace
```

The **NI** column shows how **nice** this process is. The nicer the process, the less CPU it asks. The `nice` values can be from -20 to 19. To interpret this value, look at it like this: a process with nice = **-20** is ANGRY and gets more priority for CPU and RAM while a process with nice = **19** is SUPER NICE and lets **other** processes use the resources before her).

The default value for `nice` is normally set to 0; this can be checked with:

Copy

```
$ nice
0
```

You can also check the nice-ness using the `ps` command:

Copy

```
$ ps -l
F S   UID   PID  PPID  C PRI  NI ADDR SZ WCHAN  TTY          TIME CMD
0 S  1000 15044 15035  0  80   0 -  7453 wait   pts/29   00:00:00 bash
0 S  1000 15052 15044  0  60 -20 -  3976 hrtime pts/29   00:00:00 sleep
0 R  1000 15080 15044  0  80   0 -  4680 -      pts/29   00:00:00 ps
```

Setting nice-ness

If you need to change the niceness level of a program you can run it with `nice` command and `-n` switch (for nice):

Copy

```
$ nice -n 19 echo "I am running!"
I am running!
$ nice -n -20 echo "I am running!"
nice: cannot set niceness: Permission denied
I am running!
$ sudo nice -n -20 echo "I am running!"
I am running!
```

As you can in the above example, only the root can issue high-priority niceness (below 0).

> If you run a command with `nice` without `-n`, the default will be `-n 10`

Copy

```
$ nice xeyes &
[1] 15217
$ ps -l
F S   UID   PID  PPID  C PRI  NI ADDR SZ WCHAN  TTY          TIME CMD
0 S  1000 15044 15035  0  80   0 -  7455 wait   pts/29   00:00:00 bash
0 S  1000 15217 15044  0  90  10 - 12522 poll_s pts/29   00:00:00 xeyes
0 R  1000 15218 15044  0  80   0 -  4680 -      pts/29   00:00:00 ps
```

Changing priorities

The `renice` command can be used to change the *niceness* of your running processes (or others if you are root):

Copy

```
$ ps -ef | grep firefox
jadi     13605 11226 30 08:28 ?        00:10:13 /usr/lib/firefox/firefox
jadi     15192 15044  0 09:01 pts/29   00:00:00 grep firefox
$ sudo renice -n -10 13605
13605 (process ID) old priority 5, new priority -10
```

> You can also press `r` in `top` command to renice a process

103.7 Search text files using regular expressions

Weight: 3

Candidates should be able to manipulate files and text data using regular expressions. This objective includes creating simple regular expressions containing several notational elements as well as understanding the differences between basic and extended regular expressions. It also includes using regular expression tools to perform searches through a filesystem or file content.

Objectives

- Create simple regular expressions containing several notational elements.
- Understand the differences between basic and extended regular expressions.
- Understand the concepts of special characters, character classes, quantifiers, and anchors.
- Use regular expression tools to perform searches through a filesystem or file content.

- Use regular expressions to delete, change and substitute text.

- grep

- egrep
- fgrep
- sed
- regex(7)

Regex

Regular expression, Regex, regex is a pattern to describe what you want to *match* from a text. For example `a` and `ad` both matches `jadi`. `d.` is a *deeper* example because `.` means *anything* so `d.` will match the last two characters of `jadi`. In this section, we will cover the *grep** (generalised regular expression processor) command. It has different regex *dialect*; in short Basic regex and Extended regex.

Regex basics

Simple match You can simply write down whatever you want to match and regex will search for that.

Regex	Will match
a	**a**fter, min**a**, b**a**nana, j**a**di
na	**na**rator, mi**na**, **na**nanana batman, so**na**r

Repeating

- The `\*` means repeating the previous character 0 or more times.
- The `+` means repeating the previous character 1 or more times.
- the `?` means zero or one repeats.
- `{n,m}` means the item needs to match at least n times, but not more than m times.

Regex	Will match	Note
a*b	ab, aaab, aaaaab, aaabthis	
a*b	b, sober	we should have zero or more `a`s and then a `b`
a+b	ab, aab, aaabenz	wont match **sober** or **b** because there needs to be at least one **a**
a?b	ab, a**ab**, b, batman (zero a then b), ...	

Alternation (|)

If you say `a|b` it will match `a` or `b`.

Character Classes

The dot (`.`) means any character. So `..` will match anything with at least two characters in it. You can also create your classes with [abc] which will match a or b or c and [a-z] which match a to z.

You can also refer to digits with \d.

regex is case-sensitive.

Ranges

There are shorthands for commonly used classes. Named classes to start with `[:` and end with `:]`.

Range	Meaning
[:alnum:]	Alphanumeric characters
[:blank:]	Space and tab characters
[:digit:]	The digits 0 through 9 (equivalent to 0-9)
[:upper:] and [:lower:]	Upper and lower case letters, respectively.
^ (negation)	As the first character after [in a character class negates the sense of the remaining characters

A common used regex is .* which matches any character (zero or any length).

Matching at specific locations

- The caret **^** means beginning of the string.
- The dollar **$** means the end of the string.

Samples

- `^a.*` Matches anything that starts with **a**.
- `^a.*b$` Matches anything that starts with **a** and ends with **b**.
- `^a.*\d+.*b$` Matches anything that starting with **a**, have some digits in the middle and ends with **b**.
- `^(l|b)oo` Matches anything that starts with **l** or **b** and then have **oo**
- `[f-h]|[A-K]$` The last character should be **f** to **h** (small) or **A** to **K** (capital)

grep

The `grep` command can search inside the files.

Copy

```
$ grep funk words
Garfunkel
Garfunkel's
funk
funked
funkier
funkiest
funking
funk's
funks
funky
```

These are the most common switches:

switch	meaning
-c	just show the count
-v	reverse the search
-n	show line numbers
-l	show only file names
-i	case insensitive
-r	Read all files under each directory, recursively

Copy

```
$ grep a *txt
friends.txt:Rosha
friends.txt:Xavier
friends.txt:Krishna
friends.txt:Mary
my_thinkgs.txt:laptop
$ grep z *txt
$ grep z *txt -i
friends.txt:Zee
$ grep x words -i -c
2264
$ grep z *txt -i -l
```

```
friends.txt
$ grep Z *txt -v
friends.txt:Rosha
friends.txt:Jim
friends.txt:Xavier
friends.txt:Krishna
friends.txt:Mary
my_thinkgs.txt:laptop
my_thinkgs.txt:pillow
my_thinkgs.txt:shorts
my_thinkgs.txt:t-shirt
$ grep Z friends.txt -v
Rosha
Jim
Xavier
Krishna
Mary
$
```

As another example, let's search all the `/etc` for all files containing an IP address and send the errors (mostly "you do not have permission to read this") to `/dev/null`:

Copy

```
$ egrep -r "192.168.(1|0)." /etc/ 2> /dev/null
/etc/privoxy/config:#      address 192.168.0.1 on your local private network
/etc/privoxy/config:#      (192.168.0.0) and has another outside connection with a
/etc/privoxy/config:#        listen-address  192.168.0.1:8118
/etc/avahi/hosts:# 192.168.0.1 router.local
/etc/dhcp/dhclient-exit-hooks.d/rfc3442-classless-routes:#   new_rfc3442_classless_static_routes='24 192 168 10 192 168 1 1 8 10 10 17 66 41'
/etc/dhcp/dhclient-exit-hooks.d/rfc3442-classless-routes:#   192.168.10.0/24 via 192.168.1.1
/etc/hosts:192.168.1.22 atiteltestbed
/etc/hosts:192.168.100.244 adpsms
/etc/ppp/options:# ms-dns 192.168.1.1
/etc/ppp/options:# ms-dns 192.168.1.2
/etc/ppp/options:# ms-wins 192.168.1.50
/etc/ppp/options:# ms-wins 192.168.1.51
/etc/ssl/openssl.cnf:# proxy = # set this as far as needed, e.g., http://192.168.1.1:8080
/etc/cups/cups-browsed.conf:# BrowseAllow 192.168.1.12
/etc/cups/cups-browsed.conf:# BrowseAllow 192.168.1.0/24
/etc/cups/cups-browsed.conf:# BrowseAllow 192.168.1.0/255.255.255.0
/etc/cups/cups-browsed.conf:# BrowseDeny 192.168.1.13
/etc/proxychains4.conf:## Exclude connections to 192.168.1.0/24 with port 80
/etc/proxychains4.conf:# localnet 192.168.1.0:80/255.255.255.0
/etc/proxychains4.conf:## Exclude connections to 192.168.100.0/24
/etc/proxychains4.conf:# localnet 192.168.100.0/255.255.255.0
/etc/proxychains4.conf:# localnet 192.168.0.0/255.255.0.0
/etc/proxychains4.conf:#        socks4  192.168.1.49    1080
/etc/sane.d/kodakaio.conf:#net 192.168.1.2 0x4041
/etc/sane.d/kodakaio.conf:#net 192.168.1.17 0x4067
/etc/sane.d/epsonds.conf:# net 192.168.1.123
/etc/sane.d/airscan.conf:#"Kyocera MFP Scanner" = http://192.168.1.102:9095/eSCL
/etc/sane.d/airscan.conf:#ip    = 192.168.0.1    ; blacklist by address
/etc/sane.d/airscan.conf:#ip    = 192.168.0.0/24 ; blacklist the whole subnet
/etc/sane.d/dell1600n_net.conf:#named_scanner: 192.168.0.20
/etc/sane.d/epson2.conf:# net 192.168.1.123
/etc/sane.d/magicolor.conf:# net 192.168.0.1
/etc/sane.d/saned.conf:#192.168.0.1
/etc/sane.d/saned.conf:#192.168.0.1/29
/etc/fwupd/redfish.conf:# ex: https://192.168.0.133:443
```

Extended grep

Regex is cool and `grep` is awesome so many people have tried adding to them or inventing their variants. One is GNU Extended grep. This dialect of regex, does not need much escaping and you can use it via `-E` switch or using `egrep` instead of the normal `grep`. For example, `|` in an extended regex means "or". So you can do a `egrep "a|b" words` to match anything with an `a` or a `b`.

Fixed grep

If you need to search for exact strings (and not interpret it as a regex), use `grep -F` or `fgrep` so the `fgrep this$` won't go for the end of the line and will find *this$that* instead.

sed

In previous lessons, we saw simple `sed` usage and now I have great news for you: **sed understands regex**! You can use `-r` switch to tell sed that we are using regexes.

Copy

```
$ sed -r "s/(Z|R|J)/starts with ZRJ/" friends.txt
starts with ZRJee
starts with ZRJosha
starts with ZRJim
Xavier
Krishna
Mary
```

Common switches:

switch	meaning
-r	use advanced regex
-n	suppress output, you can use p at the end of your regex (/something/p) to print the output

Copy

```
sed -rn "s/happy/HAPPY/p" words
HAPPY
slapHAPPY
unHAPPY
```

Still eager to learn? See how you can solve the Wordle using regexes:

103.8 Basic file editing

Weight: 3

Candidates should be able to edit text files using vi. This objective includes vi navigation, vi modes, inserting, editing, deleting, copying and finding text. It also includes awareness of other common editors and setting the default editor.

Objectives

- Navigate a document using vi.
- Understand and use vi modes.
- Insert, edit, delete, copy and find text in vi.
- Awareness of Emacs, nano and vim.

- Configure the standard editor.

- vi

- /, ?
- h,j,k,l
- i, o, a
- d, p, y, dd, yy
- ZZ, :w!, :q!
- EDITOR

Introduction

As any other tool, we have a wide range of selection when it comes to text editors. One of the most common and super powerful choices is the `vi` editor. It is pre-installed on all major Linux distributions and you can be sure that knowing it, will let you edit your files on all environments, let it be a remote server over SSH or a coding environment on your desktop machine or a CyberDeck machine with a minimal keyboard. Its only *drawback* might be its kind of slow learning curve but I'm sure after a 1 hour session with it, you will manage to find your way in `vi`.

There is an *Improved* version of `vi` which is called *VIMproved* or `vim`. Sometimes thats what you will find on your system and sometimes the `vi` command is aliased or linked to `vim`. Lets check this on our system (Ubuntu 22.04):

Copy

```
jadi@funlife:~$ whatis vi
vi (1)               - Vi IMproved, a programmer's text editor
jadi@funlife:~$ whereis vi
vi: /usr/bin/vi /usr/share/man/man1/vi.1.gz
jadi@funlife:~$ whatis vim
vim (1)              - Vi IMproved, a programmer's text editor
jadi@funlife:~$ whereis vim
vim: /usr/bin/vim /etc/vim /usr/share/vim /usr/share/man/man1/vim.1.gz
jadi@funlife:~$ vi --version
VIM - Vi IMproved 9.0 (2022 Jun 28, compiled Aug 23 2022 20:18:58)
Included patches: 1-242
Modified by team+vim@tracker.debian.org
Compiled by team+vim@tracker.debian.org
Huge version without GUI.  Features included (+) or not (-):
+acl               +file_in_path      +mouse_urxvt       -tag_any_white
+arabic            +find_in_path      +mouse_xterm       -tcl
+autocmd           +float             +multi_byte        +termguicolors
+autochdir         +folding           +multi_lang        +terminal
-autoservername    -footer            -mzscheme          +terminfo
-balloon_eval      +fork()            +netbeans_intg     +termresponse
+balloon_eval_term +gettext           +num64             +textobjects
-browse            -hangul_input      +packages          +textprop
++builtin_terms    +iconv             +path_extra        +timers
+byte_offset       +insert_expand     -perl              +title
+channel           +ipv6              +persistent_undo   -toolbar
+cindent           +job               +popupwin          +user_commands
-clientserver      +jumplist          +postscript        +vartabs
-clipboard         +keymap            +printer           +vertsplit
+cmdline_compl     +lambda            +profile           +vim9script
+cmdline_hist      +langmap           -python            +viminfo
+cmdline_info      +libcall           +python3           +virtualedit
+comments          +linebreak         +quickfix          +visual
+conceal           +lispindent        +reltime           +visualextra
+cryptv            +listcmds          +rightleft         +vreplace
+cscope            +localmap          -ruby              +wildignore
+cursorbind        -lua               +scrollbind        +wildmenu
+cursorshape       +menu              +signs             +windows
+dialog_con        +mksession         +smartindent       +writebackup
+diff              +modify_fname      +sodium            -X11
+digraphs          +mouse             -sound             -xfontset
-dnd               -mouseshape        +spell             -xim
-ebcdic            +mouse_dec         +startuptime       -xpm
+emacs_tags        +mouse_gpm         +statusline        -xsmp
+eval              -mouse_jsbterm     -sun_workshop      -xterm_clipboard
+ex_extra          +mouse_netterm     +syntax            -xterm_save
+extra_search      +mouse_sgr         +tag_binary
```

```
-farsi             -mouse_sysmouse    -tag_old_static
   system vimrc file: "/etc/vim/vimrc"
     user vimrc file: "$HOME/.vimrc"
 2nd user vimrc file: "~/.vim/vimrc"
      user exrc file: "$HOME/.exrc"
       defaults file: "$VIMRUNTIME/defaults.vim"
  fall-back for $VIM: "/usr/share/vim"
Compilation: gcc -c -I. -Iproto -DHAVE_CONFIG_H -Wdate-time -g -O2 -ffile-prefix-map=/build/vim-Oy69Mt/vim-9.0.0242=. -flto=auto -ffat-lto-ob
Linking: gcc -Wl,-Bsymbolic-functions -flto=auto -ffat-lto-objects -flto=auto -Wl,-z,relro -Wl,-z,now -Wl,--as-needed -o vim -lm -ltinfo -lse
```

To edit a file with vi, just give the file name to it:

Copy

```
$ vi file.txt
```

vi modes

`vi` works in two modes:

1. **Command mode** is where you go around the file, search, delete text, copy paste, replace, ... and give other commands to the vi. Some commands start with a `:` and some are only a keypress.
2. **Insert mode** is where what you type, goes into the file at the cursors position.

   To switch to the Command mode, press the ESC key. To go back to the Insert mode, you can use several commands but one common one is pressing the `i` key.

Moving the cursor

To move around a text file, use these keys in Command mode:

key	function
h	One character to the left (only current line)
j	One line down
k	One line up
l	One character to the right (only current line)
w	Next word on the current line
e	Next end of word on the current line
b	Previous beginning of the word on the current line
Ctrl-f	Scroll forward one page
Ctrl-b	Scroll backward one page

Typing a number before most commands will repeat the command that many times (i.e. `6h` will go 6 characters to the left)

Jumping around

key	function
G	With no number, will jump to the end & 10G will jump to line 10
H	5H will go to the 5th line from the top of the screen
L	3L will move the cursor to the 3rd line to the last line of the screen

Editing text

These command during the *command mode* will help you enter, edit, replace and text:

key	function
i	Enter the insert mode
a	Enter the insert mode after the current position of the cursor
r	replace only one character
o	open a new line below the cursor and go to the insert mode
O	open a new line above the cursor and go to the insert mode
c	clear to a location and go to the insert mode the replace till there and then normal insert (`cw` will overwrite the current word)
d	delete. you can mix with w (`dw`) to delete a word. Same as cw but dw does not go to the insert mode
dd	Delete the current line
x	Delete character at the position of the cursor
p	Paste the last deleted text after the cursor

key	function
P	Paste the last deleted text before the cursor
xp	swaps the character at the cursor position with the one on its right

Searching

key	function
/	Search forward (`/happiness` will find the next happiness)
?	Search backward
n	repeat previous search. You can also use `/` and `?` without any parameters)

Search wraps around to the top once the bottom of file is reached

Exiting

It is always funny when you see someone entering to the vi and not knowing how to exit! Learn these and prevent the laughter:

key	function
:q!	Quit editing without saving = runaway after any mistake
:w!	Write the file (whether modified or not). Attempt to overwrite existing files or read-only or other unwritable files
:w myfile.txt	Write to a new name
ZZ	Exit and save the file if modified
:e!	Reload the file from disk
:!	Run a shell command

Entering colon (`:`) during *command mode* will move the cursor to the bottom of the screen and vi will wait for your commands. Press ESC to return back to the normal command mode.

The exclamation mark in most commands will say "I know what I'm doing" and will write on read-only files if you have access and will exit without asking

it is possible to combine commands. For example you can combine `:w` and `:q` and just say `:wq` (write and exit).

help

You can always ask for help with `:help` or `:help subject`. This way vi will open a help text which you can use / search just like any other text. Close it with `:q` command.

Other editors

You can also use other editors if you want. One easy to use and common option is `nano` and some other choices are `micro`, `emacs` (full featured) and `neovim` (an update to vim).

Default editor

The default editor in `bash` is set using the `EDITOR` environment variable. You can change it with:

Copy

```
$ export EDITOR='vim'
```

or by adding the above line to the `.bashrc` file. We will see these in more details in next chapters.

104.1 Create partitions and filesystems

Weight: 2

Description: Candidates should be able to configure disk partitions and then create filesystems on media such as hard disks. This includes the handling of swap partitions.

Objective

- Manage MBR and GPT partition tables
- Use various mkfs commands to create various filesystems such as:
  - ext2/ext3/ext4
  - XFS
  - VFAT
  - exFAT
- Basic feature knowledge of Btrfs, including multi-device filesystems, compression and subvolumes.
- fdisk
- gdisk
- parted
- mkfs
- mkswap

Block devices

A block device is a nonvolatile mass storage device whose information can be accessed in any order; like hard disks, USB memories, floppy disks, and CD-ROMs. We *format* these devices to fixed sized blocks.

We can check all block devices using `lsblk` command. In addition on a long ls format (`-l`), block devices are shown with a `b` at the first column:

Copy

```
jadi@debianamd:~$ ls /dev/ -l | grep "^b"
brw-rw----  1 root disk      8,   0 Feb  3  2023 sda
brw-rw----  1 root disk      8,  16 Feb  3  2023 sdb
brw-rw----  1 root disk      8,  17 Feb  3  2023 sdb1
brw-rw----  1 root disk      8,  18 Feb  3  2023 sdb2
brw-rw----  1 root disk      8,  19 Feb  3  2023 sdb3
brw-rw----+ 1 root cdrom    11,   0 Feb  3  2023 sr0
```

It is possible to create **partitions** on a block device and even split it and use it as multiple disks. Systems with old BIOS boot loaders use the **Master Boot Record (MBR)** method for patitioning and newer UEFI systems, do you **GUID Parition Table (GPT)** formats.

Linux systems use `udev` to add block devices and their paritions to the `/dev` in the form of `/dev/sdb1` (2nd disk (b) and first parition (1)).

Editing Partition Tables

fdisk

`fdisk` is the main command for viewing / changing and creating partitions on MBR systems. the `-l` switch lists the partitions:

Copy

```
# fdisk -l /dev/sdb
Disk /dev/sdb: 20 GiB, 21474836480 bytes, 41943040 sectors
Disk model: QEMU HARDDISK
Units: sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
Disklabel type: gpt
Disk identifier: 11D48091-5AA7-422A-85F7-A23F476CDFD7
```

```
Device        Start      End  Sectors  Size Type
/dev/sdb1      2048  1050623  1048576  512M EFI System
/dev/sdb2   1050624 39942143 38891520 18.5G Linux filesystem
/dev/sdb3  39942144 41940991  1998848  976M Linux swap
```

- The **Boot** flag shows which partition starts the boot on DOS PCs and has no importance on LILO & GRUB
- Start and End shows where this partition is located on the disk.
- Size shows each partition size .
- ID indicates the partition format (82 is swap, 83 is linux data, ... check all with `l` in interactive mode)

It is also possible to run fdisk in interactive mode. `m` will show you the help menu:

Copy

```
~# fdisk /dev/sda

Welcome to fdisk (util-linux 2.36.1).
Changes will remain in memory only, until you decide to write them.
Be careful before using the write command.

Device does not contain a recognized partition table.
Created a new DOS disklabel with disk identifier 0xe2dbaded.

Command (m for help): m

Help:

  DOS (MBR)
   a   toggle a bootable flag
   b   edit nested BSD disklabel
   c   toggle the dos compatibility flag

  Generic
   d   delete a partition
   F   list free unpartitioned space
   l   list known partition types
   n   add a new partition
   p   print the partition table
   t   change a partition type
   v   verify the partition table
   i   print information about a partition

  Misc
   m   print this menu
   u   change display/entry units
   x   extra functionality (experts only)

  Script
   I   load disk layout from sfdisk script file
   O   dump disk layout to sfdisk script file

  Save & Exit
   w   write table to disk and exit
   q   quit without saving changes

  Create a new label
   g   create a new empty GPT partition table
   G   create a new empty SGI (IRIX) partition table
   o   create a new empty DOS partition table
   s   create a new empty Sun partition table


Command (m for help):
```

To check current partition list, try the `p` (print) command:

Copy

```
Command (m for help): p
Disk /dev/sdb: 20 GiB, 21474836480 bytes, 41943040 sectors
Disk model: QEMU HARDDISK
Units: sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
Disklabel type: gpt
Disk identifier: 11D48091-5AA7-422A-85F7-A23F476CDFD7

Device        Start      End  Sectors  Size Type
/dev/sdb1      2048  1050623  1048576  512M EFI System
```

```
/dev/sdb2   1050624 39942143 38891520 18.5G Linux filesystem
/dev/sdb3  39942144 41940991  1998848  976M Linux swap
```

You should remember the disk layouts concepts from the chapter 102.1. So lets create some paritions using `fdisk`. I will use the `n` for *new*:

Copy

```
# fdisk /dev/sda

Welcome to fdisk (util-linux 2.36.1).
Changes will remain in memory only, until you decide to write them.
Be careful before using the write command.

Device does not contain a recognized partition table.
Created a new DOS disklabel with disk identifier 0x40bd0f72.

Command (m for help): n
Partition type
   p   primary (0 primary, 0 extended, 4 free)
   e   extended (container for logical partitions)
Select (default p): p
Partition number (1-4, default 1):
First sector (2048-8388607, default 2048):
Last sector, +/-sectors or +/-size{K,M,G,T,P} (2048-8388607, default 8388607): +1G

Created a new partition 1 of type 'Linux' and of size 1 GiB.

Command (m for help): p
Disk /dev/sda: 4 GiB, 4294967296 bytes, 8388608 sectors
Disk model: QEMU HARDDISK
Units: sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
Disklabel type: dos
Disk identifier: 0x40bd0f72

Device     Boot Start     End Sectors Size Id Type
/dev/sda1        2048 2099199 2097152   1G 83 Linux
```

Lets create another Extened parition and add a Linux (83) and a Swap (82) parition there.

Copy

```
Command (m for help): n
Partition type
   p   primary (1 primary, 0 extended, 3 free)
   e   extended (container for logical partitions)
Select (default p): e
Partition number (2-4, default 2):
First sector (2099200-8388607, default 2099200):
Last sector, +/-sectors or +/-size{K,M,G,T,P} (2099200-8388607, default 8388607):

Created a new partition 2 of type 'Extended' and of size 3 GiB.

Command (m for help): p
Disk /dev/sda: 4 GiB, 4294967296 bytes, 8388608 sectors
Disk model: QEMU HARDDISK
Units: sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
Disklabel type: dos
Disk identifier: 0x40bd0f72

Device     Boot   Start     End Sectors Size Id Type
/dev/sda1          2048 2099199 2097152   1G 83 Linux
/dev/sda2       2099200 8388607 6289408   3G  5 Extended

Command (m for help): n
All space for primary partitions is in use.
Adding logical partition 5
First sector (2101248-8388607, default 2101248):
Last sector, +/-sectors or +/-size{K,M,G,T,P} (2101248-8388607, default 8388607):

Created a new partition 5 of type 'Linux' and of size 3 GiB.

Command (m for help): p
Disk /dev/sda: 4 GiB, 4294967296 bytes, 8388608 sectors
Disk model: QEMU HARDDISK
Units: sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
```

```
Disklabel type: dos
Disk identifier: 0x40bd0f72

Device     Boot   Start     End Sectors Size Id Type
/dev/sda1          2048 2099199 2097152   1G 83 Linux
/dev/sda2       2099200 8388607 6289408   3G  5 Extended
/dev/sda5       2101248 8388607 6287360   3G 83 Linux
```

Oh! I forgot to allocate space for the swap partition. Lets delete the previous one and add two new ones:

Copy

```
Command (m for help): d
Partition number (1,2,5, default 5): 5

Partition 5 has been deleted.

Command (m for help): n
All space for primary partitions is in use.
Adding logical partition 5
First sector (2101248-8388607, default 2101248):
Last sector, +/-sectors or +/-size{K,M,G,T,P} (2101248-8388607, default 8388607): +1G

Created a new partition 5 of type 'Linux' and of size 1 GiB.

Command (m for help): p
Disk /dev/sda: 4 GiB, 4294967296 bytes, 8388608 sectors
Disk model: QEMU HARDDISK
Units: sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
Disklabel type: dos
Disk identifier: 0x40bd0f72

Device     Boot   Start     End Sectors Size Id Type
/dev/sda1          2048 2099199 2097152   1G 83 Linux
/dev/sda2       2099200 8388607 6289408   3G  5 Extended
/dev/sda5       2101248 4198399 2097152   1G 83 Linux

Command (m for help): n
All space for primary partitions is in use.
Adding logical partition 6
First sector (4200448-8388607, default 4200448):
Last sector, +/-sectors or +/-size{K,M,G,T,P} (4200448-8388607, default 8388607):

Created a new partition 6 of type 'Linux' and of size 2 GiB.
```

And now, I have to change the type of the parition 6 to swap:

Copy

```
Command (m for help): t
Partition number (1,2,5,6, default 6): 6
Hex code or alias (type L to list all): L

00 Empty            24 NEC DOS          81 Minix / old Lin  bf Solaris
01 FAT12            27 Hidden NTFS Win  82 Linux swap / So  c1 DRDOS/sec (FAT-
02 XENIX root       39 Plan 9           83 Linux            c4 DRDOS/sec (FAT-
03 XENIX usr        3c PartitionMagic   84 OS/2 hidden or   c6 DRDOS/sec (FAT-
04 FAT16 <32M       40 Venix 80286      85 Linux extended   c7 Syrinx
05 Extended         41 PPC PReP Boot    86 NTFS volume set  da Non-FS data
06 FAT16            42 SFS              87 NTFS volume set  db CP/M / CTOS / .
07 HPFS/NTFS/exFAT  4d QNX4.x           88 Linux plaintext  de Dell Utility
08 AIX              4e QNX4.x 2nd part  8e Linux LVM        df BootIt
09 AIX bootable     4f QNX4.x 3rd part  93 Amoeba           e1 DOS access
0a OS/2 Boot Manag  50 OnTrack DM       94 Amoeba BBT       e3 DOS R/O
0b W95 FAT32        51 OnTrack DM6 Aux  9f BSD/OS           e4 SpeedStor
0c W95 FAT32 (LBA)  52 CP/M             a0 IBM Thinkpad hi  ea Linux extended
0e W95 FAT16 (LBA)  53 OnTrack DM6 Aux  a5 FreeBSD          eb BeOS fs
0f W95 Ext'd (LBA)  54 OnTrackDM6       a6 OpenBSD          ee GPT
10 OPUS             55 EZ-Drive         a7 NeXTSTEP         ef EFI (FAT-12/16/
11 Hidden FAT12     56 Golden Bow       a8 Darwin UFS       f0 Linux/PA-RISC b
12 Compaq diagnost  5c Priam Edisk      a9 NetBSD           f1 SpeedStor
14 Hidden FAT16 <3  61 SpeedStor        ab Darwin boot      f4 SpeedStor
16 Hidden FAT16     63 GNU HURD or Sys  af HFS / HFS+       f2 DOS secondary
17 Hidden HPFS/NTF  64 Novell Netware   b7 BSDI fs          fb VMware VMFS
18 AST SmartSleep   65 Novell Netware   b8 BSDI swap        fc VMware VMKCORE
1b Hidden W95 FAT3  70 DiskSecure Mult  bb Boot Wizard hid  fd Linux raid auto
1c Hidden W95 FAT3  75 PC/IX            bc Acronis FAT32 L  fe LANstep
1e Hidden W95 FAT1  80 Old Minix        be Solaris boot     ff BBT
```

```
Aliases:
   linux          - 83
   swap           - 82
   extended       - 05
   uefi           - EF
   raid           - FD
   lvm            - 8E
   linuxex        - 85
Hex code or alias (type L to list all): swap

Changed type of partition 'Linux' to 'Linux swap / Solaris'.

Command (m for help): p
Disk /dev/sda: 4 GiB, 4294967296 bytes, 8388608 sectors
Disk model: QEMU HARDDISK
Units: sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
Disklabel type: dos
Disk identifier: 0x40bd0f72

Device     Boot   Start     End Sectors Size Id Type
/dev/sda1          2048 2099199 2097152   1G 83 Linux
/dev/sda2       2099200 8388607 6289408   3G  5 Extended
/dev/sda5       2101248 4198399 2097152   1G 83 Linux
/dev/sda6       4200448 8388607 4188160   2G 82 Linux swap / Solaris
```

Satisfied! Let's verify and then write the results:

Copy

```
Command (m for help): v
No errors detected.
Remaining 4094 unallocated 512-byte sectors.

Command (m for help): w
The partition table has been altered.
Calling ioctl() to re-read partition table.
Syncing disks.

root@debianamd:~# fdisk -l /dev/sda
Disk /dev/sda: 4 GiB, 4294967296 bytes, 8388608 sectors
Disk model: QEMU HARDDISK
Units: sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
Disklabel type: dos
Disk identifier: 0x40bd0f72

Device     Boot   Start     End Sectors Size Id Type
/dev/sda1          2048 2099199 2097152   1G 83 Linux
/dev/sda2       2099200 8388607 6289408   3G  5 Extended
/dev/sda5       2101248 4198399 2097152   1G 83 Linux
/dev/sda6       4200448 8388607 4188160   2G 82 Linux swap / Solaris
```

gdisk

As seen on chapter 102.1, we have used `gdisk` on GPT machines. Its not that different from `fdisk`. Lets have a look at its main commands:

Copy

```
root@debianamd:~# gdisk /dev/sda
GPT fdisk (gdisk) version 1.0.6

Warning: Partition table header claims that the size of partition table
entries is 0 bytes, but this program  supports only 128-byte entries.
Adjusting accordingly, but partition table may be garbage.
Warning: Partition table header claims that the size of partition table
entries is 0 bytes, but this program  supports only 128-byte entries.
Adjusting accordingly, but partition table may be garbage.
Partition table scan:
  MBR: MBR only
  BSD: not present
  APM: not present
  GPT: not present


***************************************************************
Found invalid GPT and valid MBR; converting MBR to GPT format
in memory. THIS OPERATION IS POTENTIALLY DESTRUCTIVE! Exit by
typing 'q' if you don't want to convert your MBR partitions
```

```
to GPT format!
************************************************************

Command (? for help): ?
b   back up GPT data to a file
c   change a partition's name
d   delete a partition
i   show detailed information on a partition
l   list known partition types
n   add a new partition
o   create a new empty GUID partition table (GPT)
p   print the partition table
q   quit without saving changes
r   recovery and transformation options (experts only)
s   sort partitions
t   change a partition's type code
v   verify disk
w   write table to disk and exit
x   extra functionality (experts only)
?   print this menu
```

As you can see, the partition table have to be compatible with your BIOS/UEFI setup.

parted

`parted` is the GNU tool to edit partitions. Its main advantage is the ability to resize currently defined partitions but using it is a bit trickier than `fdisk` and `gdisk`:

Copy

```
# parted
GNU Parted 3.4
Using /dev/sda
Welcome to GNU Parted! Type 'help' to view a list of commands.
(parted) help
  align-check TYPE N                        check partition N for TYPE(min|opt) alignment
  help [COMMAND]                            print general help, or help on COMMAND
  mklabel,mktable LABEL-TYPE                create a new disklabel (partition table)
  mkpart PART-TYPE [FS-TYPE] START END      make a partition
  name NUMBER NAME                          name partition NUMBER as NAME
  print [devices|free|list,all|NUMBER]      display the partition table, available devices, free space, all found partitions, or a particular p
  quit                                      exit program
  rescue START END                          rescue a lost partition near START and END
  resizepart NUMBER END                     resize partition NUMBER
  rm NUMBER                                 delete partition NUMBER
  select DEVICE                             choose the device to edit
  disk_set FLAG STATE                       change the FLAG on selected device
  disk_toggle [FLAG]                        toggle the state of FLAG on selected device
  set NUMBER FLAG STATE                     change the FLAG on partition NUMBER
  toggle [NUMBER [FLAG]]                    toggle the state of FLAG on partition NUMBER
  unit UNIT                                 set the default unit to UNIT
  version                                   display the version number and copyright information of GNU Parted
```

hint? use gparted

The `gparted` tool is a graphical tool to manage your partitioned. It has the ability to resize partitions and is super easy to use. Its not part of the LPIC exam but its good to know about it. just in case ;) gparted.org

Formatting the partition

Filesystems

After you partitioned your block devices, you have to format them to make them usable to store files and directories. Formatting a file system, creates a map which stores the location and name of files and directories and make it possible to move files between folders, deleting them or renaming them; think of it as the index of a book.

There are several filesystems in the linux world but, these are the most commonly used ones:

Format	Description
ext2	second extended filesystem was developed to address shortcomings in the older Unix/Minix filesystem used in early versions of Linux. It has been used extensively on Linux for many years. There is no journaling in ext2, and it has largely been replaced by ext3 and more recently ext4.
ext3	ext2 + journaling, max file size is 2TB and max filesystem size is 16TB
ext4	current version of ext, max file size is 16TB and max filesystem size is 1EB (1000*1000TB)
XFS	journaling, caches to RAM, great for uninterruptible power supplies, Max file and filesystem size is 8EB
swap	Swap is used when the system needs to use more ram than what is has. It's like an extra ram on disk
VFAT	FAT32, no journaling, good for data exchange with windows, does not understand **permissions** and symbolic links
exFAT	Extended FAT. A newer version of FAT which is used mainly for extended device which should work on all machines; like USB disks
btrfs	A new high performance file system. Max file and filesize is 16 EB. Has its own form of RAID and LVM and build-in snapshots and fault tolerance and data compression on the fly.

Creating filesystems

You can format your partitions with `mkfs` command (and `mkswap` for swap). This is a front end to commands like `mkfs.ext3` for *ext3*, `mkfs.ext4` for *ext4* and `mkfs.reiserfs` for *ReiserFS*. full list of installed on your system is here:

Copy

```
# ls /sbin/mk*
/sbin/mkdosfs  /sbin/mkexfatfs  /sbin/mkfs.bfs     /sbin/mkfs.exfat  /sbin/mkfs.ext3  /sbin/mkfs.fat    /sbin/mkfs.msdos  /sbin/mkfs.vfat
/sbin/mke2fs   /sbin/mkfs   /sbin/mkfs.cramfs  /sbin/mkfs.ext2   /sbin/mkfs.ext4  /sbin/mkfs.minix  /sbin/mkfs.ntfs   /sbin/mkhomedir_helper
```

If using the `mkfs`, the main switch is `-type` (or `-t`) to specify the format:

Copy

```
# mkfs -t ext4 /dev/sda1
mke2fs 1.46.2 (28-Feb-2021)
Discarding device blocks: done
Creating filesystem with 262144 4k blocks and 65536 inodes
Filesystem UUID: 63625ecd-857a-419f-a300-12395aaad89f
Superblock backups stored on blocks:
    32768, 98304, 163840, 229376

Allocating group tables: done
Writing inode tables: done
Creating journal (8192 blocks): done
Writing superblocks and filesystem accounting information: done

root@debianamd:~# mkfs.exfat /dev/sda5
mkexfatfs 1.3.0
Creating... done.
```

```
Flushing... done.
File system created successfully.
```

If you need to assign a lable to the partition, you have to use the `-L lable_name` option. Please note that in recent system, people use UUIDs instead of labels. UUID of a disk can be viewed with:

Copy

```
# blkid /dev/sda1
/dev/sda1: UUID="63625ecd-857a-419f-a300-12395aaad89f" BLOCK_SIZE="4096" TYPE="ext4" PARTUUID="40bd0f72-01"
```

and as the last task, lets create a swap on `/dev/sda6`:

Copy

```
# mkswap /dev/sda6
Setting up swapspace version 1, size = 2 GiB (2144333824 bytes)
no label, UUID=6a59cf20-8fd6-4d86-b044-89f7bc67993b
```

and then

Copy

```
# swapon /dev/sda6
```

On chapter 14.3 we will see how we can *mount* / *unmount* these filesystems.

104.2 Maintain the integrity of filesystems

Weight: 2

Candidates should be able to maintain a standard filesystem, as well as the extra data associated with a journaling filesystem.

Objectives

- Verify the integrity of filesystems.
- Monitor free space and inodes.
- Repair simple filesystem problems.
- du
- df
- fsck
- e2fsck
- mke2fs
- tune2fs
- xfs_repair
- xfs_fsr
- xfs_db

du & df

In many cases you want to find out about the free space of a disk or find how much space a directory is using or check how many inodes left.

> The inode (index node) is a data structure in a Unix-style file system that describes a file-system object such as a file or a directory. Each inode stores the attributes and disk block locations of the object's data. File-system object attributes may include metadata (times of last change, access, modification), as well as owner and permission data. A directory is a list of inodes with their assigned names. The list includes an entry for itself, its parent, and each of its children. (wikipedia)

df

The **d**isk**f**ree command is used to find out about the free and used space on file systems.

Copy

```
jadi@funlife:~$ df -TH
Filesystem        Type      Size  Used Avail Use% Mounted on
/dev/sda2         ext4       23G   15G  7.7G  65% /
none              tmpfs     4.0K     0  4.0K   0% /sys/fs/cgroup
udev              devtmpfs  3.9G  4.0K  3.9G   1% /dev
tmpfs             tmpfs     788M  1.4M  786M   1% /run
none              tmpfs     5.0M  4.0K  5.0M   1% /run/lock
none              tmpfs     3.9G   19M  3.9G   1% /run/shm
none              tmpfs     100M   28K  100M   1% /run/user
/dev/mapper/chome ext4      243G  229G   14G  95% /home/jadi
/dev/sdb1         vfat      3.7G  7.8M  3.6G   1% /media/jadi/BA82-BECD
```

Here, the `-T` switch tells `df` to show the file system types and `-H` make numbers human readable (in powers of 1000 , for powers of 2, use `-h`).

On some filesystems (like ext2-4) we have a fixed number of inodes, so you may need to check the number of remaining inodes too. To do so, use the `-i` switch:

Copy

```
jadi@funlife:~$ df -i
Filesystem          Inodes  IUsed    IFree IUse% Mounted on
/dev/sda2          1531904 458616  1073288   30% /
none               1007533      4  1007529    1% /sys/fs/cgroup
udev               1003703    542  1003161    1% /dev
tmpfs              1007533    644  1006889    1% /run
none               1007533      3  1007530    1% /run/lock
none               1007533    162  1007371    1% /run/shm
none               1007533     33  1007500    1% /run/user
/dev/mapper/chome 16171008 269293 15901715    2% /home/jadi
/dev/sdb1                0      0        0     - /media/jadi/BA82-BECD
```

> vfat file format has no inodes; there is no owner or access rights on vfat filesystems.

du

The **d**isk**u**sage command shows the used space of **directories and files**. The common switches are:

switch	usage
-h	print sizes in powers of 1024 (e.g., 1023M)
-H	print sizes in powers of 1000 (e.g., 1.1G)
-c	show the grand total
--max-depth 2	Calculate all but show only 2 directories deep
-s	Only shows the summary and not all the directories one by one

Copy

```
jadi@funlife:~/w/lpic$ du
16    ./101
701456    ./done
701464    ./Logo/chert
704588    ./Logo
12    ./data
12    ./100
9432884    .
jadi@funlife:~/w/lpic$ du -c
16    ./101
701456    ./done
701464    ./Logo/chert
704588    ./Logo
12    ./data
12    ./100
```

```
9432884    .
9432884    total
jadi@funlife:~/w/lpic$ du -hs
9.0G    .
```

in many cases when I want to see what uses my serers space, I use something like `$sudo du /home -h --max-depth 1`

checking file systems

fsck

If anything bad happens for your filesystem (say power suddenly goes down) you will have a corrupted file system. The general command to fix this is `fsck`. Technically this command is a front end for many commands:

Copy

```
jadi@funlife:~$ ls /sbin/*fsck*
/sbin/dosfsck       /sbin/fsck.ext2     /sbin/fsck.fat     /sbin/fsck.vfat
/sbin/e2fsck       /sbin/fsck.ext3     /sbin/fsck.minix
/sbin/fsck       /sbin/fsck.ext4     /sbin/fsck.msdos
/sbin/fsck.cramfs  /sbin/fsck.ext4dev  /sbin/fsck.nfs
```

Some of these are just hardlinks to `e2fsck` command

A common switch during boot is `-A` which tells `fsck` to check all file systems in /etc/fstab ordered by *passno* in that file which is 6th field (File systems with *passno* of 0, wont be checked during the boot.

Copy

```
root@funlife:~# fsck /dev/sdb
fsck from util-linux 2.25.1
e2fsck 1.42.10 (18-May-2014)
/dev/sdb is in use.
e2fsck: Cannot continue, aborting.


root@funlife:~# umount /dev/sdb
umount: /dev/sdb: not mounted
root@funlife:~# umount /dev/sdb1
root@funlife:~# fsck /dev/sdb
fsck from util-linux 2.25.1
e2fsck 1.42.10 (18-May-2014)
ext2fs_open2: Bad magic number in super-block
fsck.ext2: Superblock invalid, trying backup blocks...
fsck.ext2: Bad magic number in super-block while trying to open /dev/sdb

The superblock could not be read or does not describe a valid ext2/ext3/ext4
filesystem.  If the device is valid and it really contains an ext2/ext3/ext4
filesystem (and not swap or ufs or something else), then the superblock
is corrupt, and you might try running e2fsck with an alternate superblock:
    e2fsck -b 8193 <device>
 or
    e2fsck -b 32768 <device>
```

You can also check filesystems with UUID (find them with `blkid` command or with labels) :

Copy

```
root@funlife:~# fsck /dev/sdb
fsck from util-linux 2.25.1
e2fsck 1.42.10 (18-May-2014)
ext2fs_open2: Bad magic number in super-block
fsck.ext2: Superblock invalid, trying backup blocks...
fsck.ext2: Bad magic number in super-block while trying to open /dev/sdb

The superblock could not be read or does not describe a valid ext2/ext3/ext4
filesystem.  If the device is valid and it really contains an ext2/ext3/ext4
filesystem (and not swap or ufs or something else), then the superblock
is corrupt, and you might try running e2fsck with an alternate superblock:
    e2fsck -b 8193 <device>
 or
    e2fsck -b 32768 <device>

root@funlife:~# blkid
/dev/sda1: LABEL="movies"
/dev/sdb1: UUID="BA82-BECD" TYPE="vfat" PARTUUID="381add66-01"
root@funlife:~# fsck LABEL=movies
fsck from util-linux 2.25.1
root@funlife:~# fsck UUID="BA82-BECD"
fsck from util-linux 2.25.1
fsck.fat 3.0.26 (2014-03-07)
/dev/sdb1: 14 files, 1972/945094 clusters
```

Use `-N` to see what command/test is going to be executed without actually running them:

Copy

```
root@funlife:~# fsck -N UUID="BA82-BECD"
fsck from util-linux 2.25.1
[/sbin/fsck.vfat (1) -- /dev/sdb1] fsck.vat /dev/sdb1
```

If you want to check a XFS filesystem, you have to use `xfs_check` command

Some version do have a `-a` for automatic fixing all found issues but its not recommended.

e2fsck

`e2fsck` is used to check the ext2/ext3/ext4 family of file systems. For ext3 and ext4 file systems that use a `journal`, if the system has been shutdown uncleanly without any errors,normally, after replaying the committed transactions in the journal, the file system should be marked as clean. Hence, for file systems that use **journaling** , e2fsck will normally replay the journal and exit, unless its superblock indicates that further checking is required.

device is a block device (e.g., `/dev/sdc1`) or file containing the file system.

Note that in general it is not safe to run `e2fsck` on ***mounted*** file systems. The only exception is if the `-n` option is specified, and `-c`, `-l`, or `-L` options are not specified. However, even if it is safe to do so, the results printed by `e2fsck` are not valid if the file system is ***mounted*** . If `e2fsck` asks whether or not you should check a file system which is mounted, the only correct answer is ``no''. Only experts who really know what they are doing should consider answering this question in any other way.

If `e2fsck` is run in interactive mode (meaning that none of `-y`, `-n`, or `-p` are specified), the program will ask the user to fix each problem found in the file system. A response of `'y'` will fix the error; `'n'` will leave the error unfixed; and `'a'` will fix the problem and all subsequent problems; pressing Enter will proceed with the default response, which is printed before the question mark. Pressing `Control-C` terminates `e2fsck` immediately.

mke2fs

mke2fs is used to create an ext2, ext3, or ext4 filesystem, usually in a disk partition. device is the special file corresponding to the device (e.g /dev/hdXX). blocks-count is the number of blocks on the device. If omitted, mke2fs automagically figures the file system size. If called as mkfs.ext3 a journal is created as if the `-j` option was specified.

The defaults of the parameters for the newly created filesystem, if not overridden by the options listed below, are controlled by the /etc/mke2fs.conf configuration file. See the mke2fs.conf(5) manual page for more details.

tune2fs

This is a command to tune *ext* file systems. It can show information and set many options. The `-l` option lists the current configs:

Copy

```
jadi@funlife:~$ sudo tune2fs -l /dev/sda2
tune2fs 1.42.10 (18-May-2014)
Filesystem volume name:   <none>
Last mounted on:          /
Filesystem UUID:          1651a94e-0b4e-47fb-aca0-f77e05714617
```

```
Filesystem magic number:  0xEF53
Filesystem revision #:    1 (dynamic)
Filesystem features:      has_journal ext_attr resize_inode dir_index filetype needs_recovery extent flex_bg sparse_super large_file huge_file
Filesystem flags:         signed_directory_hash
Default mount options:    user_xattr acl
Filesystem state:         clean
Errors behavior:          Continue
Filesystem OS type:       Linux
Inode count:              1531904
Block count:              6123046
Reserved block count:     306152
Free blocks:              2302702
Free inodes:              1073461
First block:              0
Block size:               4096
Fragment size:            4096
Reserved GDT blocks:      1022
Blocks per group:         32768
Fragments per group:      32768
Inodes per group:         8192
Inode blocks per group:   512
Flex block group size:    16
Filesystem created:       Mon Dec  1 10:21:42 2014
Last mount time:          Sat Jan 31 17:21:51 2015
Last write time:          Sat Jan 31 17:21:51 2015
Mount count:              32
Maximum mount count:      -1
Last checked:             Mon Dec  1 10:21:42 2014
Check interval:           0 (<none>)
Lifetime writes:          103 GB
Reserved blocks uid:      0 (user root)
Reserved blocks gid:      0 (group root)
First inode:              11
Inode size:              256
Required extra isize:     28
Desired extra isize:      28
Journal inode:            8
First orphan inode:       786620
Default directory hash:   half_md4
Directory Hash Seed:      16c38a41-e709-4e04-b1c2-8a79d71ea7e8
Journal backup:           inode blocks
```

debugfs

This is an interactive tool for debug an ext filesystem. It opens the filesystem in read-only mode unless we tell it not to (with `-w` option). It can un-delete files and directories..

Copy

```
root@funlife:~# debugfs /dev/sda2
debugfs 1.42.10 (18-May-2014)
debugfs:  cd /etc/        <-- cd
debugfs:  pwd            <-- show were am I
[pwd]   INODE: 524289  PATH: /etc
[root]  INODE:      2  PATH: /
debugfs:  stat passwd        <-- show data on one file
Inode: 527187   Type: regular    Mode:  0644   Flags: 0x80000
Generation: 1875144872    Version: 0x00000000:00000001
User:     0   Group:     0   Size: 2145
File ACL: 0    Directory ACL: 0
Links: 1   Blockcount: 8
Fragment:  Address: 0    Number: 0    Size: 0
 ctime: 0x548d4241:a7b196fc -- Sun Dec 14 11:24:41 2014
 atime: 0x54cc635b:6acfc148 -- Sat Jan 31 08:38:43 2015
 mtime: 0x548d4241:a01076f8 -- Sun Dec 14 11:24:41 2014
crtime: 0x548d4241:9f1c52f8 -- Sun Dec 14 11:24:41 2014
Size of extra inode fields: 28
EXTENTS:
(0):2188172
debugfs:  ncheck 527187        <-- node check an inode
Inode    Pathname
527187    /etc/passwd
debugfs:  q            <-- quit
```

Superblock

Unix systems use superblocks to save *filesystem metadata*. Most of the times this block is located at the beginning of the file system and replicated on other locations too. The `-n` of `mke2fs` displays superblock locations

Copy

```
# mke2fs -n /dev/sda7
mke2fs 1.41.9 (22-Aug-2009)
Filesystem label=
OS type: Linux
Block size=4096 (log=2)
Fragment size=4096 (log=2)
7159808 inodes, 28637862 blocks
1431893 blocks (5.00%) reserved for the super user
First data block=0
Maximum filesystem blocks=4294967296
874 block groups
32768 blocks per group, 32768 fragments per group
8192 inodes per group
Superblock backups stored on blocks:
    32768, 98304, 163840, 229376, 294912, 819200, 884736, 1605632, 2654208,
    4096000, 7962624, 11239424, 20480000, 23887872
```

xfs tools

Note: in some distros, xfs tools are not installed by default and you might need to install `xfsprogs` package.

This is same as the `tune2fs` but for xfs file systems.

> xfs_info should be used on mounted file systems

Command	usage
xfs_info	display information
xfs_growfs	expand file system
xfs_admin	change parameters on XFS file systems
xfs_repair	repair the problems. Please note that the filesytem under repair should be unmounted
xfs_db	checks and debug the filesystem. xfs_db is used to examine an XFS filesystem. Under rare circumstances it can also be used to modify an XFS filesystem, but that task is normally left to xfs_repair or to scripts such as xfs_admin that run xfs_db.
xfs_fsr	filesystem reorganizer for XFS. When invoked with no arguments xfs_fsr reorganizes all regular files in all mounted filesystems. xfs_fsr makes many cycles over /etc/mtab each time making a single pass over each XFS filesystem. Each pass goes through and selects files that have the largest number of extents. It attempts to defragment the top 10% of these files on each pass.

Repairing

We used the `fsck` for showing file system information but it is designed to *fix* file systems too. If the boot time check find a problems, you will be put into a command line to fix the problems.

On non-journaling file systems (ext2) the fsck will show you many questions about each block and you have to say `y` if you want it to fix them. On journaling file systems (ext3&4, xfs, ..) the fsck has much less tasks to perform.

> for xfs file systems, we have `xfs_check` command

An important switch is `-n` which causes these commands **not to fix** anything and just show what was going to be done.

Other tools

For the LPIC exam, it is good to know about these commands.

filesystem	command	usage
ext	tune2fs	Show or set ext2 and ext3 parameters or even set the journaling options
ext	dumpe2fs	Prints the super block and block group descriptor information for an ext2 or ext3 filesystem.
ext	debugfs	Is an interactive file system debugger. Use it to examine or change the state of an ext2 or ext3file system.
reiserfs	reiserfstune	show and set parameters
reiserfs	debugreiserfs	Prints the super block and block group descriptor information for an ext2 or ext3 filesystem.
XFS	xfs_info	display information
XFS	xfs_growfs	expand file system
XFS	xfs_admin	change parameters on XFS file systems
XFS	xfs_repair	repair the problems
XFS	xfs_db	checks and debugs the filesystem

104.3 Control mounting and unmounting of filesystems

weight: 3

Candidates should be able to configure the mounting of a filesystem.

- Manually mount and unmount filesystems.
- Configure filesystem mounting on bootup.
- Configure user mountable removable filesystems.
- Use of labels and UUIDs for identifying and mounting file systems.
- Awareness of systemd mount units.
- /etc/fstab
- /media/
- mount
- umount
- blkid
- lsblk

Mounting and Unmounting

When we have a formatted partition and need to use it, we have to `mount` it somewhere in the Linux directory hierarchy. Unlike Windows, the new *driver* do now show up as separated disks, but like *virtual subdirectories* somewhere in your `/` tree.

Say we want to *mount* the `/dev/sda3` on `/media/mydisk`. The directory `/media/mydisk` should be there and then, we just run:

Copy

```
sudo mount -t ext4 /dev/sda3 /media/mydisk
```

All files and folders in /dev/sda3 will be accessible from /media/mydisk.

Run `mount` with no parameter to see all mounted devices. To **un mount**, simply use the `umount` on the drive or the directory. These two are equivalent:

Copy

```
sudo umount /dev/sda3
sudo umount /media/mydisk
```

Mounting and umounting can happen on many different storage types, for example on NFS storages, ISO (with `-o loop`), tmpfs, ...

> swap disks do not need mounting. You should use `swapon` and `swapoff` to use them.

Copy

```
mount -t ext4 /dev/sda1 /media
```

The `-t` switch indicates the type of the filesystem .

Copy

```
mount -o remount,ro /dev/sda1
```

The `-o` switch passes some options (say `ro` for readonly)

It's usually used as one-line command:

Copy

```
mount -t ext4 -o remount,ro /dev/sda1 /media
```

> The `/media` and `/mnt` directories are used to mount filesystems, even though you can use any directory for this purpose.

UUID & Labels

As you already know, there is a problem when working with classical device names like `/dev/vdb1`: they change! The current `/dev/sdb` might be seen as `/dev/sdd` after you remove / reconnect it. To solve this, its better to work with UUIDs (Universal Unique Identifiers). Check them with `lsblk` (`-O` will show all available columns or specify with -o as below) and `blkid`.

Copy

```
# lsblk -o +UUID
NAME   MAJ:MIN RM  SIZE RO TYPE MOUNTPOINTS                              UUID
sr0     11:0    1  1.8G  0 rom  /run/media/jadi/Fedora-WS-Live-35_B-1-2 2021-09-22-21-47-34-00
zram0  251:0    0  1.9G  0 disk [SWAP]
vda    252:0    0   20G  0 disk
├─vda1 252:1    0  600M  0 part /boot/efi                                E13A-EF36
├─vda2 252:2    0    1G  0 part /boot                                    19ed96a1-3b36-4202-81bb-349f7adfb8b1
└─vda3 252:3    0 18.4G  0 part /home                                    076766a5-8864-4e35-a632-464b03396f7a
                               /
vdb    252:16   0    2G  0 disk
└─vdb1 252:17   0    2G  0 part /tmp/lkj                                4c1a51e6-47bf-4a34-84a2-87027c91e14a

# blkid
/dev/vdb1: UUID="4c1a51e6-47bf-4a34-84a2-87027c91e14a" BLOCK_SIZE="4096" TYPE="ext4" PARTUUID="5415e516-01"
/dev/sr0: BLOCK_SIZE="2048" UUID="2021-09-22-21-47-34-00" LABEL="Fedora-WS-Live-35_B-1-2" TYPE="iso9660"
/dev/zram0: LABEL="zram0" UUID="e459f522-1675-40d2-b318-51d9bd16d7bb" TYPE="swap"
/dev/vda2: UUID="19ed96a1-3b36-4202-81bb-349f7adfb8b1" BLOCK_SIZE="4096" TYPE="ext4" PARTUUID="5f4ee154-3ade-4af6-8809-6d90d5827d39"
/dev/vda3: LABEL="fedora_localhost-live" UUID="076766a5-8864-4e35-a632-464b03396f7a" UUID_SUB="a4340a29-6d9b-4c28-a7c8-b4aab5d08893" BLOCK_SI
/dev/vda1: UUID="E13A-EF36" BLOCK_SIZE="512" TYPE="vfat" PARTLABEL="EFI System Partition" PARTUUID="a7a2b260-0302-45bc-a4db-42bd2e0ee7f2"

# blkid /dev/vdb1
/dev/vdb1: UUID="4c1a51e6-47bf-4a34-84a2-87027c91e14a" BLOCK_SIZE="4096" TYPE="ext4" PARTUUID="5415e516-01"

# mount UUID="4c1a51e6-47bf-4a34-84a2-87027c91e14a" /media/mydisk/
```

fstab

For automatic mounting, Linux uses the `/etc/fstab` file. Its like a table which shows what file system should be mounted where during the boot. This is my the `/etc/fstab` of my Fedora:

Copy

```
# cat /etc/fstab

#
# /etc/fstab
```

```
# Created by anaconda on Wed Oct 20 13:16:38 2021
#
# Accessible filesystems, by reference, are maintained under '/dev/disk/'.
# See man pages fstab(5), findfs(8), mount(8) and/or blkid(8) for more info.
#
# After editing this file, run 'systemctl daemon-reload' to update systemd
# units generated from this file.
#
UUID=076766a5-8864-4e35-a632-464b03396f7a /                       btrfs   subvol=root,compress=zstd:1 0 0
UUID=19ed96a1-3b36-4202-81bb-349f7adfb8b1 /boot                   ext4    defaults        1 2
UUID=E13A-EF36          /boot/efi               vfat    umask=0077,shortname=winnt 0 2
UUID=076766a5-8864-4e35-a632-464b03396f7a /home                   btrfs   subvol=home,compress=zstd:1 0 0
```

These are the columns:

- file system: Label, UUID, device
- mount point: swap or none for swap
- type: can be ext4 , xfs ,nfs or other types of filesystem
- options: defaults, rw / ro, noauto, user, exec / noexec, noatime, umask
- dump: do dump command backup this? mostly 0
- pass: Non-zero values of pass specify the order of checking filesystems at boot time

note:

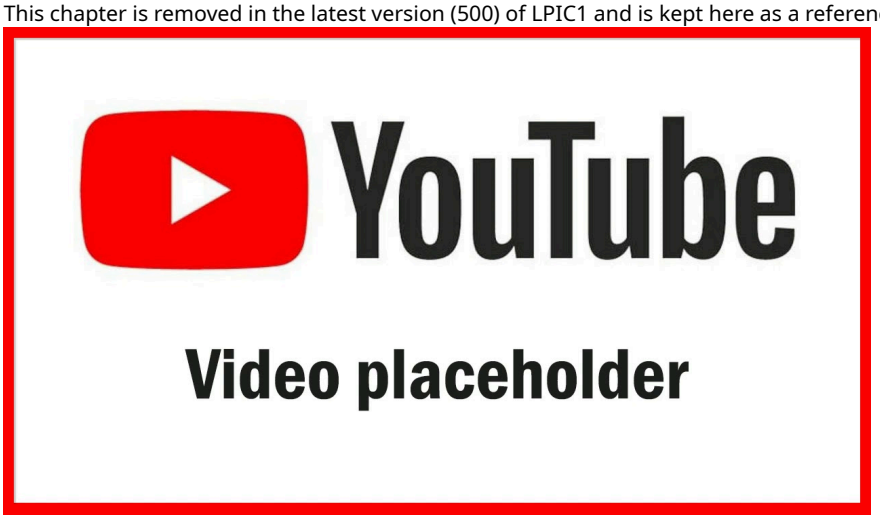
- User-mounted filesystems default to noexec unless exec is specified after user.
- noatime will disable recording of access times. Not using access times may improve performance.

Systemd mount units

When using systemd, a unit configuration file whose name ends in ".mount" encodes information about a file system mount point controlled and supervised by systemd.

104.4 Removed

This chapter is removed in the latest version (500) of LPIC1 and is kept here as a reference.



104.4 Manage disk quotas

Weight: 1

Candidates should be able to manage disk quotas for users.

Objectives

- Set up a disk quota for a filesystem.
- Edit, check and generate user quota reports.
- quota
- edquota
- repquota

- quotaon

Enabling quotas

Quotas will let the system admin to control how much a user or a group consumes disk. The version 2 quota discussed in LPIC, needs kernel 2.4 and above. The package is called `quota`.

The option should be added to required `/etc/fstab` file. The most famous ones are:

option	meaning
usrquota	user quotas
uqouta	same as usrquota
quota	same as usrquota
grpquota	group quotas
gquota	same as grpquota

So for example is we want to enable quotas on sda2 we have to change the line in /etc/fstab like this:

Copy

```
/dev/sda2  /home  ext4    defaults,usrquota,grpquota          1 2
```

Next we need to specify the quotas of each user and each group. Two files called `aquota.user` and `aquota.group` in the root file system will do this. Now it is enough to run the `quotacheck` command.

> the `quotacheck` command will create the aquota.user and aquota.group if they do not exist

Copy

```
# quotacheck -u -a -m -c -v
quotacheck: Your kernel probably supports journaled quota but you are not using it. Consider switching to journaled quota to avoid running quo
quotacheck: Scanning /dev/sda1 [/boot] done
quotacheck: Old group file name could not been determined. Usage will not be subtracted.
quotacheck: Checked 13 directories and 389 files
# ls /boot/
aquota.user
```

Creates quota files for **u**sers on **a**ll file systems and will work on **m**ounted file systems; being **v**erbose.

Then you need to turn the quota checking on:

Copy

```
# quotaon -auv ##all in /etc/fstab, for user quotas and be verbose
/dev/sda1 [/boot]: user quotas turned on
```

Setting limits

The main command for *editing* quota is **edquota*. It will check the users quota from all file systems and presents them in a file editor to you.

Copy

```
#edquota -u jadi
Disk quotas for user jadi (uid 1000):
  Filesystem                   blocks       soft       hard     inodes     soft     hard
  /dev/sda1                         0          0          0          0        0        0
```

As you can see, the system shows the current blocks of 1k data, number of inodes (number of files and directories) and soft and hard limits for each of them. If a user goes over its soft-limits, there will be emails. Hard limits are real limits and user can not go over them. If you need to save soft or hard limits, just change the file and save it.

> You have to run `quotacheck` to update these data

For copying one users limits to another user, use the `-p` switch:

Copy

```
# edquota -p jadi newuser neweruser lastuser
```

quota reports

If you need to check the quota of only one user use the `quota` command.

Copy

```
# quota  jadi
Disk quotas for user jadi (uid 1000):
     Filesystem  blocks   quota   limit   grace   files   quota   limit   grace
      /dev/sda1       5    5000       0               2       0       0
```

This is not easy if you have many users so you can use `repquota` as follow:

Copy

```
# repquota -u -a
*** Report for user quotas on device /dev/sda1
Block grace time: 7days; Inode grace time: 7days
                        Block limits                File limits
User            used    soft    hard  grace    used  soft  hard  grace
----------------------------------------------------------------------
root      --  120288       0       0            401     0     0
jadi      --       5    5000       0              2     0     0
```

Warning users

There is a command for checking quotas and warning users called `warnquota`. It will be good to run it time to time using a crontab (will see this crontabs later).

.

.

.

.

.

.

.

.

.

.

.

104.5 Manage file permissions and ownership

Weight: 3

Candidates should be able to control file access through the proper use of permissions and ownerships.

Key Knowledge Areas

- Manage access permissions on regular and special files as well as directories.
- Use access modes such as suid, sgid and the sticky bit to maintain security.
- Know how to change the file creation mask.
- Use the group field to grant file access to group members.
- chmod
- umask
- chown
- chgrp

Users and Groups

A linux system can have many users and many groups. You can login with one user and use `su` command to change to another group. Each user belongs to one primary group and can be a member of other groups too.

To check your user and group use the `whoami`, `groups` and `id` commands.

Copy

```
- $ whoami
jadi
- $ groups
jadi adm cdrom sudo dip plugdev netdev lpadmin sambashare debian-tor
- $ id
uid=1000(jadi) gid=1000(jadi) groups=1000(jadi),4(adm),24(cdrom),27(sudo),30(dip),46(plugdev),102(netdev),108(lpadmin),124(sambashare),125(del
- $ su root -
Password:
- # id
uid=0(root) gid=0(root) groups=0(root)
- # exit
exit
- $ whoami
jadi
```

As you can see `id` shows both user and group information.

> Note the 0 UID and 0 GID belong to the root user and root group.

These are stored in `/etc/passwd` and `/etc/group` files.

Copy

```
$ cat /etc/group | grep adm
adm:x:4:syslog,jadi
lpadmin:x:108:jadi
```

File ownership & permissions

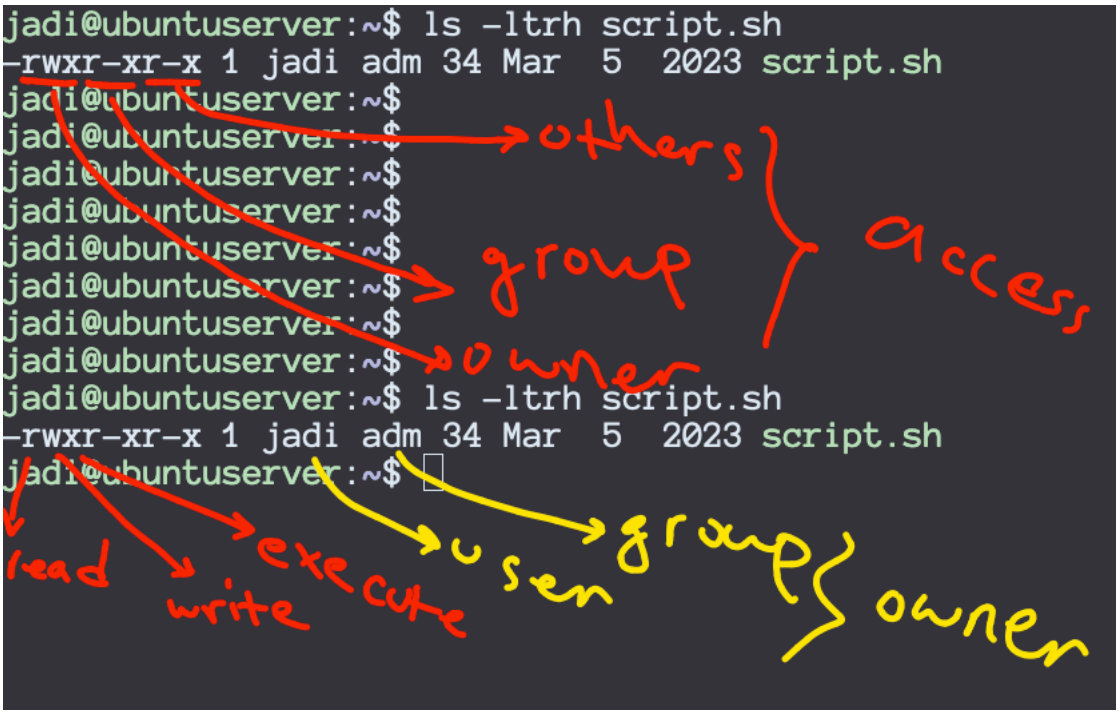
Linux uses three layers of access/permissions for each file or directory: User, Group & Others.

Each file belongs to one user and one group and this user and the members of that group will have specific read/write/execute accesses on it. Have a look:

Copy

```
$ ls -ltrh script.sh
-rwxr-xr-x 1 jadi adm 34 Mar  5  2023 script.sh
```

In above example, `jadi` is the owner of the file. The file belongs to the `adm` group and the owner (`jadi` here) has **r**ead, **w**rite (including deletion and edit) & **e**xecute (reading directory content) permissions on the file while the `adm` group members and **others** only have read & execute access.



> In many distros, when you create a user, the system creates a group with the same name and assigns that user's files to that group.

The below table shows some more information regarding the first part of the `ls -l` command:

Position	Meaning
1	What this entry is. Dash (-) is for ordinary files, 'l' is for links & 'd' is for directory
2,3,4	read, write and execute access for the owner
5,6,7	read, write and execute access for the group members
8,9,10	read, write and execute access for other users
11	Indicated if any other access methods (such as SELinux) are applies to this file - not part of the LPIC 101 exam

Lets check another example.

Copy

```
$ ls -l /sbin/fdisk
-rwxr-xr-x 1 root root 267176 Oct 15 18:58 /sbin/fdisk
```

We can see that the `fdisk` can be read, be written and be executed by its owner (root). Other users (even if they belong to the group `root`) can only read and execute it.

> although non-root users can execute the fdisk, this program wont do much if it sees that a non root user is running it.

Lets look at another example:

Copy

```
$ ls -l /home/
total 12
drwxr-xr-x 160 jadi jadi 12288 Feb  7 11:44 jadi
```

The first character is a `d` so this is a directory. The owner (`jadi`) has read, write and execute access but other members of the `jadi` **group** and **others** only have read and execute access on this directory.

execute (`x` character) means that they can see the files inside it, have a look if dir has not `x`:

Copy

```
(with root user)
$ tree /tmp/FOO
FOO
├── Bar1
└── Bar2

$ ls -ld /tmp/FOO/
drw-r--r--. 2 jadi jadi 80 Mar 23 19:32 FOO/

(With other user)
$ ls -l FOO/
ls: cannot access 'FOO/Bar2': Permission denied
ls: cannot access 'FOO/Bar1': Permission denied
total 0
-????????? ? ? ? ?            ? Bar1
-????????? ? ? ? ?            ? Bar2
```

Changing permissions

It is possible to change the permissions on files & directories using the `chmod` command. There are two ways to tell this command what you want to do:

1. using octal (base 8) codes
2. using short codes

When using octal codes, you have to to create an octal number to tell chmod what you want to do. In this method, `0` means no access, `1` means execute, `2` means write and `4` means read. So if you want to give ***read***+***execute***, you have to give `4+1` which is `5`. The below table shows every possible combination:

Symbolic	Octal
rwx	7
rw-	6
r-x	5
r--	4
-wx	3
-w-	2

Symbolic	Octal
--x	1
---	0

So if you want to give `rwx` to the owner, `rx` to the group and only `x` to others, you have to use `751`:

Copy

```
$ ls -ltrh myfile
-rw-rw-r-- 1 jadi jadi 0 Feb  8 21:01 myfile
$ chmod 751 myfile
$ ls -ltrh myfile
-rwxr-x--x 1 jadi jadi 0 Feb  8 21:01 myfile
```

This might look difficult but there are some commonly used combinations like `755` for general executable files or `600` for personal files.

There is also an *easier* method. In this method `u` means user, `g` means group and `o` means others. You can append `+x` to give execute permission, `+r` to give read permission and `+w` to give write permission. For example `u+x` will grant execute permission to user. If you want to remove a permission, use a `-` sign. For example `g-r` to prevent `group` members from `read`ing the file.

Copy

```
$ ls -ltrh myfile
-rwxr-x--x 1 jadi jadi 0 Feb  8 21:01 myfile
$ chmod u-x myfile
$ ls -ltrh myfile
-rw-r-x--x 1 jadi jadi 0 Feb  8 21:01 myfile
$ chmod +x myfile
$ chmod uo+xr myfile
$ ls -ltrh myfile
-rwxr-xr-x 1 jadi jadi 0 Feb  8 21:01 myfile
```

One very common switch on `chmod` is `-R` for recursive `chmod`ing on files. This will give `read` permission of all files inside /tmp/ to any user:

Copy

```
# chmod -R o+r /tmp
```

Changing owner and groups

If you need to change the ownership or group of a file or directory, use the `chown` command:

Copy

```
$ ls -ltrh newfile
-rw-rw-r-- 1 jadi jadi 0 Feb  8 21:38 newfile
$ chown root:root newfile
chown: changing ownership of ‘newfile’: Operation not permitted
$ sudo chown root:root newfile
[sudo] password for jadi:
$ ls -ltrh newfile
-rw-rw-r-- 1 root root 0 Feb  8 21:38 newfile
```

A common switch is `-R` to `chown` recursively.

If need to only change the group you may use `chgrp` command:

Copy

```
$ sudo chgrp postgres newfile
$ ls -ltrh newfile
-rw-rw-r-- 1 root postgres 0 Feb  8 21:38 newfile
```

It is possible to assign more groups to a user via `usermod` command:

Copy

```
$ sudo usermod -aG sudo jadi # will add jadi to the sudo group
```

Since users can be a member of many groups, they might need to change their **default group** when creating files. To do so, check the groups with `groups` command and set the default one with `newgrp`:

Copy

```
$ touch newfile
$ ls -ltrh newfile
-rw------- 1 jadi jadi 0 Feb  8 21:53 newfile
$ groups
```

```
jadi adm cdrom sudo dip plugdev netdev lpadmin sambashare debian-tor
$ newgrp adm
$ touch newerfile
$ ls -ltrh new*
-rw------- 1 jadi jadi 0 Feb  8 21:53 newfile
-rw------- 1 jadi adm  0 Feb  8 21:54 newerfile
```

Access modes

A question for you: If we do not have write access to `/etc/passwd` or `/etc/shadow`, how is it possible to change our password then?

Normally when you run a program, it runs with *your* access level. But what happens if you need to run a program with a higher access level? say to do changes in the password files? For this Linux sets two special bits for each file; **suid** (set user id) and **sgid** (set group id). If these bits are set on a file, that file will be executed with the access of the **owner** (or **group**) of the file and not the user who is running it.

Copy

```
$ ls -ltrh /usr/bin/passwd
-rwsr-xr-x 1 root root 50K Jul 18  2014 /usr/bin/passwd
```

Kindly note the `s` in the *executable bit* for the user permissions and also for the group? That means when any user runs this program, it will be run with the owner of the file access level (which is root) instead of that user's id.

It is possible to set/unset the `suid` and `sgid` using `chmod` and `+s` or `-s` instead of `x`.

While we are on this, let me introduce you to the last special bit. It is called **sticky bit** and if set, makes the file resistant to ***deletion***. If the sticky bit is set, ONLY the owner of the file will be able to delete it, even if others do have write access to it. This is useful for places like /tmp (everybody has write access on /tmp directory but we do not want others to delete our files).

Sticky bit is identified by `t` and will be shown on the last bit of a `ls -l`:

Copy

```
$ ls -dl /tmp
drwxrwxrwt 13 root root 77824 Feb  8 21:27 /tmp
```

As you can see the sticky bit is set and although all users have write access in this directory, they wont be able to delete each others files.

Lets review how you can set these access modes:

access mode	octal	symbolic
suid	4000	u+s
sgid	2000	g+s
sticky	1000	+t

guid on a directory will force any new file in that directory to have the guid of the directory itself.

umask

Another tricky question: what is the permissions of a newly created file? What happens if you `touch` a non-existing file? What will be its permissions? This is set by the `umask`. This command tells the system what permissions (in addition to execute) **should not be given to** new files. In other words, imagine that all new files will have `666` octal permission (write + read for user, group & others), so a umask of 0002 (yes! 4 digits) will remove the 2 (write) for others (666-0002=664):

Copy

```
$ umask
0002
```

If we need to change umask, it can be done with the same command:

Copy

```
$ umask
0002
$ touch newfile
$ ls -ltrh newfile
-rw-rw-r-- 1 jadi jadi 0 Feb  8 21:38 newfile
$ mkdir newdir
$ ls -ltrhd newdir
drwxrwxr-x 2 jadi jadi 4.0K Feb  8 21:38 newdir
$ umask u=rw,g=,o=
$ touch newerfile
$ ls -l newerfile
-rw------- 1 jadi jadi 0 Feb  8 21:41 newerfile
```

```
$ umask
0177
```

Note how I used `u=rw,g=,o=` to tell `umask` or `chmod` what I exactly wanted from it.

104.6 Create and change hard and symbolic links

weight: 2

Candidates should be able to create and manage hard and symbolic links to a file.

Key Knowledge Areas

- Create links.
- Identify hard and/or softlinks.
- Copying versus linking files.
- Use links to support system administration tasks.
- `ln`
- `unlink`

links

On a storage device, a file or directory is saved on some block and a reference to it is saved in the FAT, ext, ... allocation table alongside data about its owner, permissions, when it was last accessed, its size and such. As you already know, we can check this using the `ls` command.

Copy

```
$ ls -i script.sh
785379 script.sh
$ ls -l script.sh
-rwxr--r-t 1 jadi adm 34 Mar  5 13:38 script.sh
$ ls -R
.:
check_tr181_file.php  etc_apparmom.tar  index.html  php_checker  script.sh  test.csv  w

./php_checker:
check_tr181_file.php  index.html  test.csv

./w:
```

But it is also possible to create links. A link is simply an additional directory entry for a file or directory:

Copy

```
$ ls -l
total 4
-rwxr-xr-x 1 jadi adm 34 Mar  5 13:38 script.sh
$ ln -s script.sh copy_of_script.sh
$ ls -ltrh
total 4.0K
-rwxr-xr-x 1 jadi adm  34 Mar  5 13:38 script.sh
lrwxrwxrwx 1 jadi jadi  9 Mar  6 11:31 copy_of_script.sh -> script.sh
$ ls -i
785349 copy_of_script.sh  785379 script.sh
```

A link is simply an additional directory entry for a file or directory, allowing two or more names for the same file. A **hard link** is a directory entry that points to an inode, while a **soft link or symbolic link** is a directory entry that points to an inode that provides the name of another directory entry. The exact mechanism for storing the second name may depend on both the file system and the length of the name. Symbolic links are also called symlinks.

Soft links, or symlinks, merely point to another file or directory by name rather than by inode. Soft links can cross file system boundaries.

Copy

```
$ vi new_file
$ ls -ltrh
total 4.0K
-rw------- 1 jadi jadi 9 Mar  6 11:37 new_file
```

```
$ ln new_file hard_link
$ ln -s new_file soft_link
$ ls -ltrh
total 8.0K
-rw------- 2 jadi jadi 9 Mar  6 11:37 new_file
-rw------- 2 jadi jadi 9 Mar  6 11:37 hard_link
lrwxrwxrwx 1 jadi jadi 8 Mar  6 11:37 soft_link -> new_file
$ rm new_file
$ ls -ltrh
total 4.0K
-rw------- 1 jadi jadi 9 Mar  6 11:37 hard_link
lrwxrwxrwx 1 jadi jadi 8 Mar  6 11:37 soft_link -> new_file
$ cat hard_link
new file
$ cat soft_link
cat: soft_link: No such file or directory
```

> Note the `l` as the first character in `ls -l`'s permission list when we have a symbolic link.

You can create hard links only for files, not for directories. The exception is the special directory entries in a directory for the directory itself and for its parent (`.` and `..`)

> You will get an error if you attempt to create hard links that cross file systems or that are for directories.

Copy

```
$ ln mydir link2dir # ln: mydir: hard link not allowed for director
$ ln -s mydir link2dir # works just fine
```

If you are using relative names, you will usually want the current working directory to be the directory where you are creating the link; otherwise, the link you create will be relative to another point in the file system.

Copy

```
$ ln -s myfile.txt mydir/ #broken link
$ cd mydir
$ ln -s ../myfile.txt .
```

It's recommended to use exact path for links.

Copy

```
$ ln -s /tmp/myfile.txt /tmp/Foo/Bar/myfile.txt
$ ls -l /tmp/Foo/Bar/myfile.txt
lrwxrwxrwx. 1 jadi jadi 9 Mar  6 12:37 /tmp/Foo/Bar/myfile.txt -> /tmp/myfile.txt
```

we can find symbolic links using the `find` command:

Copy

```
$ find . -type l
```

They are highly used to keep one specific name pointing to a changing binary name. For example we always need to be able to run `python3` so we point it to the latest installed python on the system:

Copy

```
$ which python3
/usr/bin/python3
$ ls -l /usr/bin/python3
lrwxrwxrwx 1 root root 10 Mar 25  2022 /usr/bin/python3 -> python3.10
```

To remove links, you can use the `rm` or `unlink` command.

104.7 Find system files and place files in the correct location

Weight: 2

Candidates should be thoroughly familiar with the **Filesystem Hierarchy Standard (FHS)**, including typical file locations and directory classifications.

Key Knowledge Areas

- Understand the correct locations of files under the FHS.
- Find files and commands on a Linux system.

- Know the location and purpose of important file and directories as defined in the FHS.

- `find`

- `locate`
- `updatedb`
- `whereis`
- `which`
- `type`
- `/etc/updatedb.conf`

FHS

Filesystem Hierarchy Standard (FHS) is a document describing the Linux/Unix file hierarchy. It is very useful to know these because it lets you easily find what you are looking for as a system admin.

directory	usage
/	Primary hierarchy root and root directory of the entire file system hierarchy
/bin	Essential command binaries
/boot	Static files of the boot loader
/dev	Device files
/etc	Host-specific system configuration
/lib	Essential shared libraries and kernel modules
/media	Mount point for removable media
/mnt	Mount point for mounting a filesystem temporarily
/opt	Add-on application software packages
/sbin	Essential system binaries
/srv	Data for services provided by this system
/tmp	Temporary files
/usr	Secondary hierarchy
/var	Variable data
/home	User home directories (optional)
/lib	Alternate format essential shared libraries (optional)
/root	Home directory for the root user (optional)

The `/usr` is the second level of the hierarchy. It containins shareable, read-only data. It can be shared between systems, although present practice does not often do this.

The `/var` filesystem contains variable data files, including spool directories and files, administrative and logging data, and transient and temporary files. Some portions of /var are not shareable between different systems, but others, such as /var/mail, /var/cache/man, /var/cache/fonts, and /var/spool/news, may be shared.

Path

A general linux install has a lot of files; 741341 files in my case. So how the shell finds and runs a command? This is done by a variable called PATH:

Copy

```
$ echo $PATH
/home/jadi/bin:/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin:/usr/games:/usr/local/games;/home/jadi/bin/
```

And for the root user:

Copy

```
# echo $PATH
/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin
```

As you can see, this is the list of directories separated with colon. Obviously you can change your path with `export PATH=$PATH:/usr/new/dir` or put this in `.bashrc` to make it permanent.

Locating files

`which`, `type` and `whereis`

The `which` command shows the first appearance of the command given in the path. In other words the `which mkfs` will tell you what will be run if you issue this `mkfs` command.

Copy

```
$ which mkfs
/usr/bin/mkfs
jadi@ubuntuserver:~$ which ping
/usr/bin/ping
jadi@ubuntuserver:~$ which -a ping
/usr/bin/ping
/bin/ping
```

> use the `-a` switch to show all appearance in the path and not only the first one.

But where is the `cd` command?

Copy

```
jadi@ubuntuserver:~$ which cd
jadi@ubuntuserver:~$ type cd
cd is a shell builtin
```

As you can see, `which` did not find anything for `cd`, so we tried it with `type` to see what the h**l it is.

Copy

```
$ type type
type is a shell builtin
$ type for
for is a shell keyword
$ type mkfs
mkfs is /sbin/mkfs
$ type chert
bash: type: chert: not found
```

The `type` command is more general than `which` and also understand and shows the *bash keywords*.

Another useful command in this category is `whereis`. Unlike `which`, `whereis` shows man pages and source codes of programs alongside their binary location.

Copy

```
$ whereis mkfs
mkfs: /usr/sbin/mkfs /usr/share/man/man8/mkfs.8.gz
```

> tip: there is also a `whatis` command, try it.

find

We have already seen this command in chapter 103.7 but let's see a couple of new switches.

- The `-user` and `-group` specifies a specific user & group
- The `-maxdepth` tells the find how deep it should go into the directories.

Copy

```
$ find /tmp/ -maxdepth 1 -user jadi | head
/tmp/1
/tmp/2
```

Or even find the files not belonging to any user/group with `-nouser` and `-nogroup`.

> Like other *tests*, you can add a `!` just before any phrase to negate it. So this will find files **not belonging** to jadi: `find . ! -user jadi`

It is also very common to use for files with specific strings in their names:

Copy

```
$ sudo find /etc -iname "*vmware*"
/etc/vmware-tools
/etc/vmware-tools/scripts/vmware
```

locate & updatedb

You tries `find` and loved it but there is an issue with it: it does a live active search! This can slow down your system or push too much pressure on your hard or on larger file systems take too long. To solve this, there is a faster command:

Copy

```
$ locate networking
/etc/cloud/cloud.cfg.d/subiquity-disable-cloudinit-networking.cfg
/snap/core20/1408/usr/lib/python3/dist-packages/cloudinit/distros/networking.py
/snap/core20/1408/usr/lib/python3/dist-packages/cloudinit/distros/__pycache__/networking.cpython-38.pyc
/snap/core20/1826/usr/lib/python3/dist-packages/cloudinit/distros/networking.py
/snap/core20/1826/usr/lib/python3/dist-packages/cloudinit/distros/__pycache__/networking.cpython-38.pyc
/usr/lib/python3/dist-packages/cloudinit/distros/networking.py
/usr/lib/python3/dist-packages/cloudinit/distros/__pycache__/networking.cpython-310.pyc
/usr/lib/python3/dist-packages/sos/report/plugins/networking.py
/usr/lib/python3/dist-packages/sos/report/plugins/__pycache__/networking.cpython-310.pyc
```

And it is fast:

Copy

```
$ time locate kernel / | wc -l
16989

real    0m0.091s
user    0m0.094s
sys 0m0.034s
```

This is fast because its data comes from a database created with `updatedb` (its package is called `plocate` on debian). Usually this command runs automatically (with a cronjob) on a daily basis. Its configuration file is `/etc/updatedb.conf` or `/etc/sysconfig/locate`:

Copy

```
$ cat /etc/updatedb.conf
PRUNE_BIND_MOUNTS="yes"
# PRUNENAMES=".git .bzr .hg .svn"
PRUNEPATHS="/tmp /var/spool /media /home/.ecryptfs"
PRUNEFS="NFS nfs nfs4 rpc_pipefs afs binfmt_misc proc smbfs autofs iso9660 ncpfs coda devpts ftpfs devfs mfs shfs sysfs cifs lustre tmpfs usb
```

You can update the db by running `updatedb` as root.

105.1 Customize and use the shell environment

Weight: 4

Candidates should be able to customize shell environments to meet users' needs. Candidates should be able to modify global and user profiles.

Key Knowledge Areas

- Set environment variables (e.g. PATH) at login or when spawning a new shell
- Write Bash functions for frequently used sequences of commands
- Maintain skeleton directories for new user accounts
- Set command search path with the proper directory

Terms

- `.`
- `source`
- `/etc/bash.bashrc`
- `/etc/profile`
- `env`
- `export`
- `set`
- `unset`
- `~/.bash\_profile`
- `~/.bash\_login`
- `~/.profile`
- `~/.bashrc`

- `~/.bash\_logout`
- `function`
- `alias`

Environment variables

Shell Environment Variables are special variables that contain information and configuration about the shell. System shell uses them when it needs to show you a prompt, run a command, or search what to run. Different operating system distributions have different environment variables and you can add yours if you need so; or want so!

env

can set, remove or display variables or even run a command in a modified environment.

Copy

```
Syntax
     env [OPTION]... [NAME=VALUE]... [COMMAND [ARGS]...]

Options

  -u NAME
  --unset=NAME
       Remove variable NAME from the environment, if it was in the
       environment.

  -
  -i
  --ignore-environment
       Start with an empty environment, ignoring the inherited
       environment.
```

You can also use the more general `printenv` command.

set

`set` allows you to change the values of shell options (variables) or to display the names and values of shell variables.

Using `set` we can configure how bash works. These are some samples:

switch	result
-b	Cause the status of terminated background jobs to be reported immediately, rather than before printing the next primary prompt.
-e	return in case a pipline, command, ... return non-zero
-n	Read commands but do not execute them; this may be used to check a script for syntax errors. This option is ignored by interactive shells.
-t	Exit after reading and executing one command.
-C	Prevent output redirection using ‘>’, ‘>&’, and ‘<>’ from overwriting existing files.

setting environment variables

Bash environment variables can contain only letters (a to z or A to Z), numbers (0 to 9) or the underscore character _ and cannot start with numbers. Traditionally we use UPPERCASE letters in our variable names.

To set a new environment variable do as follow:

Copy

```
$ name=jadi
$ desc='A programmer who enjoys cycling and promotes freedom'
$ echo $name
jadi
$ echo $desc
A programmer who enjoys cycling and promotes freedom
```

Notes:

1. There should be no spaces around `=`.
2. Use the quotes or double-quotes when your value do have spaces (or other special characters) in it.
3. Use a dollar sign only when referring to a variable, and not when defining it.

unset

This command unsets a variable or a function.

Copy

```
$ name=jadi
$ echo $name
jadi
$ unset name
$ echo $name
```

export

When setting environment variables with a simple `=`, the variable is only available in the current shell. So even if you run a new command from the current shell the variable wont be valid there (because technically the new command will be run in a sub-shell). If you want your variable to be valid in the current shell and all its sub-shells, you have to `export` it.

Copy

```
$ export name=jadi
$ echo $name
jadi
$ bash
$ echo $name
jadi
```

. (and source)

Yes only a dot! This is a shortcut for the bash source command. You can find it in files like `/etc/profile`. It runs the executable in front of it as part of the current environment (and not in a sub-process).

> Note: If you just execute a file (without source or dot) bash creates a child, runs the executable there and then closes it.

The `source` command is commonly used when you want to load new/updated environment variables or functions from a script.

Aliases

This is a way to define... um.. and `alias`. You can find them in your `~/.bashrc`:

Copy

```
alias ll='ls -alF'
alias la='ls -A'
alias l='ls -CF'
```

It is a way to define a shortcut and you can add yours:

Copy

```
alias testnet='ping 4.2.2.4'
```

functions

Like larger programming languages, Bash has functions, though in a somewhat limited implementation. A function is a subroutine, a code block that implements a set of operations, a "black box" that performs a specified task. Wherever there is repetitive code, when a task repeats with only

slight variations in procedure, then consider using a function.

Copy

```
funnyls () {
    ls -ltrh
    echo "This is a funny ls"
}
```

Different shell envs

The shell (here bash) can be started in 3 different ways:

1. When you *login* into the shell, say after a ssh or when you site behind a terminal and login into a Linux machine. This is an interactive login session
2. When you open a new terminal (say from a GUI env). This is also interactive but not a *login* shell.
3. When a shell *spawns* when you run a command. Yes.. technically when you run a new command from a shell, a sup-shell starts, runs the commands and then returns back to your shell. This is called "non interactive" shell.

We will see how different default environment variables can be set in each of these modes.

login shell

This is when you give your user and pass to enter a shell. In this case, these steps will define your shell environment:

1- `/etc/profile` is run

2- A line in `/etc/profile` runs whatever is in `/etc/profile.d/`

Now the global profile is loaded and system will go for user specific profiles:

3.1- `/home/USERNAME/.bash_profile`

3.2- `/home/USERNAME/.bash_login`

3.3- `/home/USERNAME/.profile`

> Note that only one of the 3, 4 & 5 will run. The system will go for `.bash_profile` and IF IT IS NOT THERE will try `.bash_login` and IF IT IS NOT THERE will try to run `.profile`. If any of these exists, the system wont look any further. So if you have only 4 & 5, only the 4 will be run.

At the end, the system loads:

4- `/home/USERNAME/.bashrc`

Commonly you will add whatever personal config you want into the `~/.bashrc`.

Interactive (non-login) shell

if you run a shell in an interactive mode (non-login) shell from a GUI or within another temrinal, only two file will handle the environment:

1. `/etc/bash.bashrc` (or `/etc/bashrc` in some systems)
2. `/home/USERNAME/.bashrc`

Non Interactive Shell

There is no specific file to be used in this case. Instead we have the `BASH_ENV`. When a new non-interactive shell starts, it looks into this variable and if it points to a file, runs it.

> On most Linux distributions, this environment value is not set by default.

A few more files

/etc/skel

This directory contains files which will be used as a starting template for each new user.

.bash_logout

This runs when you logout from a login shell. In many distros it only clears the screen so the next person will not be able to watch what you were doing before you logout.

105.2 Customize or write simple scripts

Weight: 4

Description

Candidates should be able to customize existing scripts, or write simple new Bash scripts.

Key Knowledge Areas:

- Use standard sh syntax (loops, tests).
- Use command substitution.
- Test return values for success or failure or other information provided by a command.
- Execute chained commands.
- Perform conditional mailing to the superuser.
- Correctly select the script interpreter through the shebang (`#!`) line.
- Manage the location, ownership, execution and suid-rights of scripts.

Terms and Utilities

- `for`
- `while`
- `test`
- `if`
- `read`
- `seq`
- `exec`
- `||`
- `&&`

combining commands

If you want to run more than one command in oneline, separate them by a `;`. So the `cd /tmp; ls` will run the `cd /tmp` and then `ls`. But there are more advanced usages too.

You can use `&&` (as logical And) and `||` (as logical OR) chaining. In case of And, the execution will stop as soon as the first one fails to execute. In case of Or, the next command will only run if the first one fails. Just like in logic gates.

Confusing? Let me explain again. The system will always try to *evaluate* the outcome of your chain. So if you have `A && B` then A fails, the system does not need to test B (because the overall result will be False anyway). Same logic works for `A || B || C`. If A works fine, the overall evaluation will be True so no need for testing B or C. But if A fails, system will try B and if B fails, we have to try C.

We use chaining to create logical flows. For example:

Copy

```
$ cp backup.gzip /backups/ && rm backup.gzip
```

works like this: copy the backup to this location and delete the file ONLY IF the previous copy was successful.

Shell Scripts

We can combine shell commands or programs and or some logic or loops to write large or small scripts. Shell scripts are useful when you want to *automate* some tasks in one larger command. Say a shell script called "do_backup.sh" may compress some files into a file, rename this file based on the current date and time and then save it on a remote site and at last delete the ones older than 1 month.

Shebang

There is a line at the beginning of scripts which starts with `#!` and continues with an executable who needs to run this script. It is called shebang and as mentioned, tells the shell which *interpreter* must be used to run this script.

> Note: In many programming languages (including Bash & Python), a `#` at the beginning of a line in script indicates *comments*. Do not confuse it with the *Shebang* (`#!`)

Commonly we run shell scripts using `#!/bin/bash` or `#!/bin/sh`

The rest of a shell script can use most of the commands you already know in addition with some more "programming" specific commands like loops, tests and such. Here is a primitive sample:

Copy

```
#!/bin/bash

echo
echo "We are learning! Wowww..."
echo
```

> The `sh` is a more basic shell but it is compatible with most of the things we talk about. There are also other options like `zsh` and `csh` but the LPIC is based on `bash`.

Variables

Already seen in the previous section. You can define variables like this `VARNAME=VALUE`. Here is a sample:

Copy

```
#!/bin/bash

NAME=Jadi

echo
echo "$NAME is learning! Wowww..."
echo
```

Note: you can also do NAME="Jadi the geeking guy" if you need to have spaces in your values

If you want to access the command line arguments in your shell script, use `$1`, `$2`, ... You can find the number of command line arguments via `$#` variable.

Command substitution

Sometimes you need to save the output of a command in a variable or use it in some way. To do so you can use `$(command)` or simply `command` (Backtick). Look at these two samples:

Copy

```
➜  FILES=$(ls -1) # the $FILES variable contains the list of all files

➜  ~ date +'%Y%m%d-%H%m'
20230408-1604
➜  ~ touch backup_`date +'%Y%m%d-%H%m'`.tar
➜  ~ ls
backup_20230408-1604.tar
```

executing scripts

In the Unix/Linux world, file should be executable to be executable! :D To do so use the `chmod` command with `+x`. After this step you can run your script by giving its path to shell. Please note that Linux by default does not looks into the current directory so if you are going to run a script at the current directory, you have to run `./script_name.sh`.

Another way to run a script is running `sh` or `bash` with the shell scripts name as its argument. That is `sh my_script.sh`.

In both above methods, the scripts runs inside a child bash process and *returns* back afterwards to the same shell. If you want to override this and *replace* your current shell with the program you are going to run, use the `exec` build-in command. The `-c` switch will run the command in a clean environment.

Conditions

Up to now, we were just running commands one by one. That is not very *programmatic*. If we are going to have some *logic* in our programs, we need *conditions* and *loops*. First we will cover conditions, using the `if` command. Its usage is like this:

Copy

```
if [condition]
then
   do something
   do another thing
else
   do new things
```

```
   even funnier things
fi
```

Note: else part is optional, `if`, `then`, `fi` is enough.

Conditions can be TRUE or FALSE. A very simple conditions is `if [ "Linux" = "Linux" ]`. Silly? I know but wait; we are learning the syntax only. Please give special attention to the *spaces* and `=` for checking if two strings are equal.

Copy

```
#!/bin/bash

kernel=$(uname -s)
if [ $kernel = "Linux" ]
then
    echo YES. You are using a Linux
else
    echo "Not a linux :("
fi
```

The command to check conditions is the `test` command but since it is used a lot, we have a shortcut for it. Instead of `test condition` you can write `[ condition ]`.

conditions	what is means
"a" = "b"	if two strings are equal (here it will return False)
"a" != "b"	string a *is not equal* to string b
4 -lt 40	if 4 is *lower than* 40 (True)
5 -gt 15	if 5 is *greater than* 15 (False)
5 -ge 5	if 5 is *greater or equal* 5
5 -le 3	if 5 is *lower or equal* to 3
9 -ne 2	9 is *not equal* with 2 (True)
-f FILENAME	if file FILENAME exists
-s FILENAME	if file exists and its size is more than 0 (Zero)
-x FILENAME	if file exists and is executable

read

Using `read` we can read the user input. Look at this:

Copy

```
#!/bin/sh

echo "what is your name?"
read NAME

echo "Hello $NAME"

if [ $NAME = "Jadi" ]
then
    echo "Oh I know you!"
else
    echo "I wish I knew you"
fi
echo "Bye"
```

You can timeout the waiting using the `-t` and show a prompt using `-p`.

Copy

```
if read -t 10 -p "Server address?" SERVER
then
    echo "Connecting to the $SERVER ..."
else
    echo
    echo "Too late!"
fi
```

loops

In programming loops are used to repeat part of the programs. We have 2 different loops in Bash; `for` and `while`. The for loops lets us run part of a program for a specific number of iterations. The `while` loop is used when we want to repeat part of a program *while* a specific condition became true.

for

The syntax is like this:

Copy

```
for VAR in SOME_LIST;
do
  some stuff with $VAR
  some other stuff
done
```

Note: the `in`, `;`, `do` and `done`.

On each iteration, the VAR will be equal to one of the SOME_LIST elements. SOME_LIST can be numbers, name of files, words, ...

Copy

```
for NUM in 1 2 3 4 5 6;
do
    echo $NUM
done
```

But what if you needed to run from 1 to 42? We have the `seq` command for that. The `seq 1 42` or its shorthand `{1..42}` will let us run a loop 42 times.

We can also use non-numeric variables here. This is a common use case:

Copy

```
for FILE in $(ls);
do
    echo $FILE
    wc -l $FILE
done
```

while

This is the syntax:

Copy

```
while [condition]
do
    do something
    do anohter thing
done
```

If your condition stays true all the time, the `while` loop will run *forever*. This is called an *infinite loop* and should be stopped by a `Ctrl+C`.

This is sample:

Copy

```
VAR=52

while [ $VAR -gt 42 ]
do
    echo VAR is $VAR and it is still greater than 42
    let VAR=VAR-1
done
```

Note the `let` usage! If you just say `VAR=1` and then `VAR=$VAR+1`, then VAR will be equal to `1+1` as an string!.

mailing the root user

For sending mail, you need to install `mailutils`. Then the `mail` command will send emails. You can send the mail to the root user by issuing this command:

Copy

```
jadi@funlife:~$ mail root
Cc:
Subject: Hi there root
hello there. This is my mail
```

And root will get this email. She can read it using `mail` command.

If you need to send emails in a script, just do:

Copy

```
$ echo "Body!" | mail -s "Subject" root
```

returned values

In the Unix world, the programs return values when they are finished. If you have programmed C, this is the `return 0` at the end. Commonly a 0 means successful execution. This return value can be read / examined using `$?` variable.

Copy

```
jadi@ubuntuserver:/etc/skel$ touch /chert
touch: cannot touch '/chert': Permission denied
$ echo $?
1
$ touch /tmp/11
$ echo $?
0
$ test -e /
$ echo $?
0
$ test -e /nonexist
$ echo $?
1
$ dummycommand
dummycommand: command not found
$ echo $?
127
```

To return values from your shell scripts, use the `exit` command; say `exit 0` for a successful exit.

Bonus

Just found my old `general_backup.sh` script and uploaded it to github for you. Nothing fancy but shows you how a general bash script can help you on your daily tasks.

105.3 Removed!

How luckY! this is removed. If you insist this used to be the old text:

105.3 SQL data management

This chapter is still a Work In Progress. Do not rely on it for LPIC version 500 exam. Will be updated in a few weeks.

Weight: 2

Candidates should be able to query databases and manipulate data using basic SQL commands. This objective includes performing queries involving joining of 2 tables and/or subselects.

Key Knowledge Areas

- Use of basic SQL commands
- Perform basic data manipulation

Terms and Utilities

- insert
- update
- select
- delete
- from
- where
- group by
- order by
- join

Databases

This module is about SQL language and MySQL is one of the many SQL databases. For this lesson, a database consists of some **table**s and each table has some **row**s and **filed**s. Lets have a look. In this lesson we are not going to *create* or *design* databases. You only need to have a general understanding of databases (SQL databases) and know some command to use (read query or update or add to them). The database I'm going to use in this lesson is called `lpic` and has two tables `contact` and `info`.

mysql command line

As I said, we are not goint to learn the `mysql` here, we only need to focus on `SQL` as a query language. You only need to know that `mysql` is a command line program to ineteractivly connect to a `mysql-server`. I use it like this:

Copy

```
$ mysql -u root -p
```

which means I'm going to use `u`ser root and will provide a password. It was also possible to say:

Copy

```
$ mysql -u root -p mypass lpic
```

to provide the pass on command line (not a good idea for security reasons!) and tell mysql program to connect to `lpic` database when it starts.

using a database

When you connect to a database, you have to use the `use` command to select which database you are going to issue commands on. Normally a database server (say mysql) can have 100s of different databases in it, each for one user or program.

Copy

```
jadi@funlife:~$ mysql -u root -p
Enter password:
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 17
Server version: 5.6.25-0ubuntu0.15.04.1 (Ubuntu)

Copyright (c) 2000, 2015, Oracle and/or its affiliates. All rights reserved.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql> SHOW DATABASES;
+--------------------+
| Database           |
+--------------------+
| information_schema |
| bad                |
| good               |
| lpic               |
| mysql              |
| performance_schema |
| ugly               |
+--------------------+
7 rows in set (0.00 sec)

mysql> USE LPIC;
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A

Database changed
mysql> SHOW TABLES;
+----------------+
| Tables_in_lpic |
+----------------+
| info           |
| phonebook      |
+----------------+
2 rows in set (0.00 sec)
```

As you can see `mysql` is friendly and shows lovely tables! I've told her to `use lpic` and then `show tables` and now I know that I have two tables: info & phonebook.

> Note: it is common to type MYSQL commands in CAPITAL LETTERS and names and values and .. in lower case.

SELECT

`SELECt` is obvious! It selects from a table. When we are not sure what field we are looking for, we can select `*` to get all fields.

Copy

```
mysql> SELECT * FROM phonebook;
+--------+------------------+----------------+
| name   | email            | phone          |
+--------+------------------+----------------+
| jadi   | jadi@jadi.net    | +9890something |
| nasrin | nasrin@lpic.test | +9898989898    |
| sina   | far@from.here    | +687randomnum  |
| haale  |                  | 0935secret     |
+--------+------------------+----------------+
4 rows in set (0.00 sec)
```

or ask for a specific field:

Copy

```
mysql> SELECT name FROM phonebook;
+--------+
| name   |
+--------+
| jadi   |
| nasrin |
| sina   |
| haale  |
+--------+
3 rows in set (0.00 sec)
```

WHERE

You can add *conditions* to your SQL queries using `WHERE`. Lets have a look at the other table we have:

Copy

```
mysql> SELECT * FROM info;
+--------+--------+--------+-------+
| name   | height | weight | mood  |
+--------+--------+--------+-------+
| jadi   |    180 |     74 | happy |
| sina   |    175 |     81 | happy |
| nasrin |    174 |     68 | happy |
| mina   |    171 |     59 | sad   |
+--------+--------+--------+-------+
4 rows in set (0.00 sec)
```

What if we only wanted to see our *happy* friends?

Copy

```
mysql> SELECT * FROM info WHERE mood = 'happy';
+--------+--------+--------+-------+
| name   | height | weight | mood  |
+--------+--------+--------+-------+
| jadi   |    180 |     74 | happy |
| sina   |    175 |     81 | happy |
| nasrin |    174 |     68 | happy |
+--------+--------+--------+-------+
3 rows in set (0.00 sec)

mysql>
```

Dont be afraid for *mina*, we will make her happy later but for now we need to see the friends who are happy and more than 80Kg.

Copy

```
mysql> SELECT * FROM info WHERE mood = 'happy' AND weight >= 80;
+--------+--------+--------+-------+
| name   | height | weight | mood  |
+--------+--------+--------+-------+
| sina   |    175 |     81 | happy |
+--------+--------+--------+-------+
3 rows in set (0.00 sec)
```

Or if I only needed the name:

Copy

```
mysql> SELECT name FROM info WHERE mood = 'happy' AND weight >= 80;
+------+
| name |
+------+
| sina |
+------+
1 row in set (0.00 sec)
```

ORDER BY

This is used if you want to **sort** the data based on one field. Here I'm checking my phone book based on peoples names:

Copy

```
mysql> SELECT * FROM phonebook ORDER BY name;
+--------+------------------+----------------+
| name   | email            | phone          |
+--------+------------------+----------------+
| haale  |                  | 0935secret     |
| jadi   | jadi@jadi.net    | +9890something |
| nasrin | nasrin@lpic.test | +9898989898    |
| sina   | far@from.here    | +687randomnum  |
+--------+------------------+----------------+
4 rows in set (0.00 sec)
```

This order can be done on any field, including numbers:

Copy

```
mysql> SELECT * FROM info ORDER BY height;
+--------+--------+--------+-------+
| name   | height | weight | mood  |
+--------+--------+--------+-------+
| mina   |    171 |     59 | sad   |
| nasrin |    174 |     68 | happy |
| sina   |    175 |     81 | happy |
| jadi   |    180 |     74 | happy |
+--------+--------+--------+-------+
4 rows in set (0.00 sec)
```

GROUP BY

This will *group* the output. Unfortunately this is not very clear. Lets see the first example:

Copy

```
mysql> SELECT * FROM info GROUP BY mood;
+------+--------+--------+-------+
| name | height | weight | mood  |
+------+--------+--------+-------+
| jadi |    180 |     74 | happy |
| mina |    171 |     59 | sad   |
+------+--------+--------+-------+
2 rows in set (0.00 sec)
```

We've seen that we have only two `mood`s in our table: sad & happy. When `SELECT`ing all fields (that is `*`) from this table `GROUP BY` mood, SQL will check all the moods, shows us only ONE from each. This can be used like the `uniq` command you leaned from LPIC101:

Copy

```
mysql> SELECT mood FROM info GROUP BY mood;
+-------+
| mood  |
+-------+
| happy |
| sad   |
+-------+
2 rows in set (0.00 sec)
```

Which gives you all available moods in the table. In real life this is not very useful and most of the times it is combined with `count`. Have a look:

Copy

```
mysql> SELECT count(mood), mood FROM info GROUP BY mood;
+-------------+-------+
| count(mood) | mood  |
+-------------+-------+
|           3 | happy |
|           1 | sad   |
```

```
+-------------+-------+
2 rows in set (0.00 sec)
```

Whic counts home many rows have that specific mood. So I have 3 happy friends and one sad friend.

> Note: `count` is not part of LPIC 105.3

INSERT

Another clear command. It adds a new row to a talbe. Say I want to add some data to phonebook:

Copy

```
mysql>  INSERT INTO phonebook (name, phone, email) VALUES ('ghasem', '+982112345678', '');
Query OK, 1 row affected (0.01 sec)

mysql> SELECT * FROM phonebook;
+--------+------------------+----------------+
| name   | email            | phone          |
+--------+------------------+----------------+
| jadi   | jadi@jadi.net    | +9890something |
| nasrin | nasrin@lpic.test | +9898989898    |
| sina   | far@from.here    | +687randomnum  |
| haale  |                  | 0935secret     |
| ghasem |                  | +982112345678  |
+--------+------------------+----------------+
5 rows in set (0.00 sec)
```

DELETE

You know it! This will DELETE from a table. But be careful of what you delete... WHERE is your fiend here:

mysql> DELETE FROM phonebook WHERE name = 'ghasem'; Query OK, 1 row affected (0.01 sec)

Copy

```
mysql> SELECT * FROM phonebook;
+--------+------------------+----------------+
| name   | email            | phone          |
+--------+------------------+----------------+
| jadi   | jadi@jadi.net    | +9890something |
| nasrin | nasrin@lpic.test | +9898989898    |
| sina   | far@from.here    | +687randomnum  |
| haale  |                  | 0935secret     |
+--------+------------------+----------------+
4 rows in set (0.00 sec)
```

UPDATE

Did I tell you that SQL looks like plain English? I was right because you know what `UPDATE` does! It updates (changes) row and again `WHERE` is your friend:

Copy

```
mysql> SELECT * FROM phonebook;
+--------+------------------+----------------+
| name   | email            | phone          |
+--------+------------------+----------------+
| jadi   | jadi@jadi.net    | +9890something |
| nasrin | nasrin@lpic.test | +9898989898    |
| sina   | far@from.here    | +687randomnum  |
| haale  |                  | 0935secret     |
+--------+------------------+----------------+
4 rows in set (0.00 sec)

mysql> UPDATE phonebook SET  email='haale@lpic.fake' WHERE name = 'haale';
Query OK, 1 row affected (0.01 sec)
Rows matched: 1  Changed: 1  Warnings: 0

mysql> SELECT * FROM phonebook;
+--------+------------------+----------------+
| name   | email            | phone          |
+--------+------------------+----------------+
| jadi   | jadi@jadi.net    | +9890something |
| nasrin | nasrin@lpic.test | +9898989898    |
| sina   | far@from.here    | +687randomnum  |
| haale  | haale@lpic.fake  | 0935secret     |
+--------+------------------+----------------+
4 rows in set (0.00 sec)
```

JOIN

The JOIN command can be complicated but luckily we are on LPIC1-Exam 102 and we do not need to learn much ;) Join will join/mix two tables. Just check this:

Copy

```
mysql>  SELECT * FROM phonebook JOIN info;
+--------+------------------+----------------+--------+--------+--------+-------+
| name   | email            | phone          | name   | height | weight | mood  |
+--------+------------------+----------------+--------+--------+--------+-------+
| jadi   | jadi@jadi.net    | +9890something | jadi   |    180 |     74 | happy |
| nasrin | nasrin@lpic.test | +9898989898    | jadi   |    180 |     74 | happy |
| sina   | far@from.here    | +687randomnum  | jadi   |    180 |     74 | happy |
| haale  | haale@lpic.fake  | 0935secret     | jadi   |    180 |     74 | happy |
| jadi   | jadi@jadi.net    | +9890something | sina   |    175 |     81 | happy |
| nasrin | nasrin@lpic.test | +9898989898    | sina   |    175 |     81 | happy |
| sina   | far@from.here    | +687randomnum  | sina   |    175 |     81 | happy |
| haale  | haale@lpic.fake  | 0935secret     | sina   |    175 |     81 | happy |
| jadi   | jadi@jadi.net    | +9890something | nasrin |    174 |     68 | happy |
| nasrin | nasrin@lpic.test | +9898989898    | nasrin |    174 |     68 | happy |
| sina   | far@from.here    | +687randomnum  | nasrin |    174 |     68 | happy |
| haale  | haale@lpic.fake  | 0935secret     | nasrin |    174 |     68 | happy |
| jadi   | jadi@jadi.net    | +9890something | mina   |    171 |     59 | sad   |
| nasrin | nasrin@lpic.test | +9898989898    | mina   |    171 |     59 | sad   |
| sina   | far@from.here    | +687randomnum  | mina   |    171 |     59 | sad   |
| haale  | haale@lpic.fake  | 0935secret     | mina   |    171 |     59 | sad   |
+--------+------------------+----------------+--------+--------+--------+-------+
16 rows in set (0.00 sec)
```

Every single row from first table (phonebook) is copied in front of the second table (info). Not very useful *yet*. It becomes useful when you give a *common field* or tell it to JOIN tables based on a criteria; using `WHERE`. Here is the magic:

Copy

```
mysql> SELECT * FROM phonebook JOIN info ON phonebook.name = info.name;
+--------+------------------+----------------+--------+--------+--------+-------+
| name   | email            | phone          | name   | height | weight | mood  |
+--------+------------------+----------------+--------+--------+--------+-------+
| jadi   | jadi@jadi.net    | +9890something | jadi   |    180 |     74 | happy |
| sina   | far@from.here    | +687randomnum  | sina   |    175 |     81 | happy |
| nasrin | nasrin@lpic.test | +9898989898    | nasrin |    174 |     68 | happy |
+--------+------------------+----------------+--------+--------+--------+-------+
3 rows in set (0.00 sec)
```

Great! Now I have my firneds list, their moods and their phone numbers! Say I'm bored and I need to phone a cool friend:

Copy

```
mysql> SELECT phonebook.name, phone, mood FROM phonebook JOIN info ON phonebook.name = info.name WHERE mood = 'happy';
+--------+----------------+-------+
| name   | phone          | mood  |
+--------+----------------+-------+
| jadi   | +9890something | happy |
| sina   | +687randomnum  | happy |
| nasrin | +9898989898    | happy |
+--------+----------------+-------+
3 rows in set (0.00 sec)
```

> Note: both tables have a field called `name` so I needed to use phonebook.name to tell SQL which name I want to show.

Obviously we can add more criteria and go out with a person shorter than 175cm:

Copy

```
mysql> SELECT phonebook.name, phone, mood FROM phonebook JOIN info ON phonebook.name = info.name AND height < 175;
+--------+-------------+-------+
| name   | phone       | mood  |
+--------+-------------+-------+
| nasrin | +9898989898 | happy |
+--------+-------------+-------+
1 row in set (0.00 sec)
```

cool? but we are not finished yet. I do not like having sad friends and I have one, lets make her happy too!

Copy

```
mysql> SELECT * FROM info WHERE mood = 'sad';
+------+--------+--------+------+
| name | height | weight | mood |
```

```
+------+--------+--------+------+
| mina |    171 |     59 | sad  |
+------+--------+--------+------+
1 row in set (0.01 sec)

mysql> UPDATE info SET mood = 'happy' WHERE name = 'mina';
Query OK, 0 rows affected (0.02 sec)
Rows matched: 1  Changed: 0  Warnings: 0

mysql> SELECT * FROM info WHERE mood = 'sad';
Empty set (0.00 sec)

mysql> SELECT * FROM info;
+--------+--------+--------+-------+
| name   | height | weight | mood  |
+--------+--------+--------+-------+
| jadi   |    180 |     74 | happy |
| sina   |    175 |     81 | happy |
| nasrin |    174 |     68 | happy |
| mina   |    171 |     59 | happy |
+--------+--------+--------+-------+
4 rows in set (0.00 sec)
```

Easy. The last command is even easier:

quit

Copy

```
mysql> quit
Bye
jadi@funlife:~$
```

.

.

.

.

.

.

.

.

.

.

106.1 Install and configure X11

Weight: 2

Candidates should be able to install and configure X11.

Key Knowledge Areas

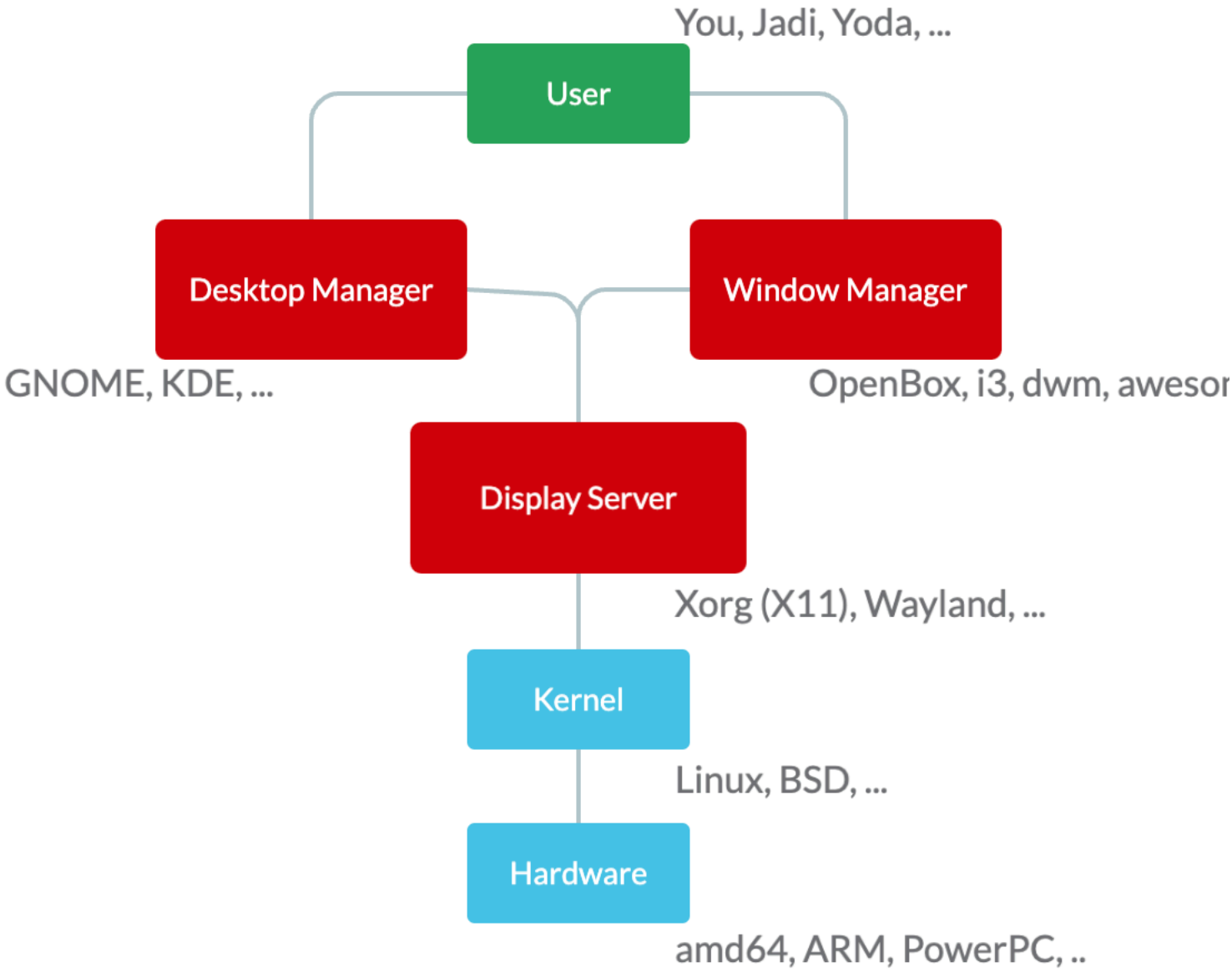
- Understanding of the X11 architecture.
- Basic understanding and knowledge of the X Window configuration file.
- Overwrite specific aspects of Xorg configuration, such as keyboard layout.
- Understand the components of desktop environments, such as display managers and window managers.
- Manage access to the X server and display applications on remote X servers.
- Awareness of Wayland.

Terms and Utilities:

- `/etc/X11/xorg.conf`
- `/etc/X11/xorg.conf.d/`
- `~/.xsession-errors`
- xhost
- xauth
- DISPLAY
- X

Intro

Many people prefer to navigate and use their system via a Graphical User Interface (Or GUI in short). A GUI based operating system provides the user the Icons, mouse, windows & folders, a metaphor from our daily "desktops". But how does this happen? As many other things in the Unix world, this happens via layers of applications that are doing one thing well. This is a rough representation of this stack:



On the lower level we have the hardware (say your Monitor) and then the Linux kernel and its drivers. On top of that there is a software called "Display Manager" or "Display Server". This program can ask the operating system (the kernel) to perform what the desktop manager requests. In other words, the display server accepts the desires of UI (even via a network channel) and translates them to the language that Kernel (and drivers in it) understand.

In this section we are focusing on **X** which is one of the two main Display Servers and then will have a brief look at the other one; the modern **Wayland**.

X

The X Window System is a network transparent window system which runs on a wide range of computing and graphics machines; including in most GNU/Linux distributions when they need a graphical interfaces. Its history a bit long and started with *XFree86* then the X.org started its own X Server which is called X11 nowadays.

> Going down the rabbit hole, *X Window System* (or *X*), was the default windowing system for most Unix and Unix like systems during 1980s. X.Org server is the free and open-source implementation of the X Window System display server by the X.Org Foundation. These day when we say *X* we are referring to the whole family of the network protocols describing how messages are exchanged between a client (application) and the display server; X11 being the 11th version of them.

/etc/X11/xorg.conf

This used to be main configuration file for X. In recent days X configures itself when starting and does not need a config file. But if you want to make changes in the boot-up process, you can create a new `xorg.conf.new` file by running the following command:

Copy

```
Xorg -configure
```

and then moving it to the `/etc/X11/xorg.conf`. This file contains different sections. Lets have a look at some of them.

Copy

```
Section "Files"
    FontPath    "/usr/share/X11/fonts/misc"
    FontPath    "/usr/share/X11/fonts/100dpi/:unscaled"
    FontPath    "/usr/share/X11/fonts/75dpi/:unscaled"
    FontPath    "/usr/share/X11/fonts/Type1"
    FontPath    "/usr/share/X11/fonts/100dpi"
    FontPath    "/usr/share/X11/fonts/75dpi"
    FontPath    "/var/lib/defoma/x-ttcidfont-conf.d/dirs/TrueType"
EndSection
```

This part is about Fonts. When X-Server is running it needs these files. FontPaths tell X11 where fonts are. It also can refer to an IP running a font-server which is not common these days. Font servers used to be responsible of rendering fonts to be shown on clients but nowadays computers are fast and can render their own fonts. Font servers are going out of fashion!

Copy

```
Section "Module"
    Load    "bitmap"
    Load    "ddc"
    Load    "dri"
    Load    "extmod"
    Load    "freetype"
    Load    "glx"
    Load    "int10"
    Load    "type1"
    Load    "vbe"
    Load    "dbe"
EndSection
```

These are modules. For example `glx` takes care of *3D graphical* effects and we are asking X to load it alongside others.

Here are the `InputDevice`s:

Copy

```
Section "InputDevice"
    Identifier    "Generic Keyboard"
    Driver        "kbd"
    Option        "CoreKeyboard"
    Option        "XkbRules"    "xorg"
    Option        "XkbModel"    "pc105"
    Option        "XkbLayout"    "us"
EndSection

Section "InputDevice"
    Identifier    "Configured Mouse"
    Driver        "mouse"
    Option        "CorePointer"
    Option        "Device"        "/dev/input/mice"
    Option        "Protocol"        "ImPS/2"
    Option        "Emulate3Buttons"    "true"
    Option        "ZAxisMapping"        "4 5"
EndSection

Section "InputDevice"
        Identifier      "Synaptics Touchpad"
        Driver          "synaptics"
        Option          "SendCoreEvents"        "true"
        Option          "Device"                "/dev/psaux"
        Option          "Protocol"              "auto-dev"
        Option        "RightEdge"        "5000"
EndSection
```

As you can see each device has an `Identifier`, `Driver` and some `options`. Above we defined a mouse, a keyboard and a touchpad and gave them some names.

Copy

```
Section "Device"
    Identifier    "ATI Technologies, Inc. Radeon Mobility 7500 (M7 LW)"
    Driver        "radeon"
    BusID        "PCI:1:0:0"
```

```
    Option        "DynamicClocks"    "on"
    Option        "CRT2HSync"    "30-80"
    Option        "CRT2VRefresh"    "59-75"
      Option        "MetaModes"    "1024x768 800x600 640x480 1024x768+1280x1024"
EndSection
```

A graphic card! Again it has its identifies (name), its drivers and some options (like support resolutions, refresh rates, ...). This device needs a screen and a monitor:

> Note: The `vesa` points to a low resolution, always working driver. It is used for troubleshooting.

Copy

```
Section "Monitor"
    Identifier    "Generic Monitor"
    Option        "DPMS"
EndSection

Section "Screen"
    Identifier    "Screen0"
    Device        "Screen0 ATI Technologies, Inc. Radeon Mobility 7500 (M7 LW)"
    Monitor        "Generic Monitor"
    DefaultDepth    24
    SubSection "Display"
        Depth        1
        Modes        "1024x768"
    EndSubSection
    SubSection "Display"
        Depth        4
        Modes        "1024x768"
    EndSubSection
    SubSection "Display"
        Depth        8
        Modes        "1024x768"
    EndSubSection
    SubSection "Display"
        Depth        15
        Modes        "1024x768"
    EndSubSection
    SubSection "Display"
        Depth        16
        Modes        "1024x768"
    EndSubSection
    SubSection "Display"
        Depth        24
        Modes        "1024x768"
    EndSubSection
EndSection
```

Note how the screen uses the already defined monitor (using its identifier "Generic Monitor") and an already defined graphic card. Also note the different color modes (say `24bit 1024x768`).

At the end we have to glue all of the above in one place as `ServerLayout`:

Copy

```
Section "ServerLayout"
    Identifier    "DefaultLayout"
    Screen        "Default Screen"
    InputDevice    "Generic Keyboard"
    InputDevice    "Configured Mouse"
    InputDevice    "Synaptics Touchpad"
EndSection
```

We have a layout with a screen and 3 input devices :)

> Note: Do not panic. A general understanding of xorg.conf is enough

/etc/X11/xorg.conf.d/

What happens if you edit the `/etc/program.conf` and then update the system? will the system overwrite your newly edited `program.conf` with the latest version from the vendor? or omit the changes in the vendor in favor of your local edits? Both are *bad*.

To solve this issue, many programs are creating a new configurations directory as `/etc/program/program.conf.d/` and asking you to add your *local* configurations there. So the main configuration file from the vendor will be `/etc/program/program.conf` and all your new configurations will go into the `/etc/program/program.conf.d/` as separated files. Much easier to manage (because smaller atomic files per configuration) and no touching the vendors config by locals. Cool? yes... this is happening in X11 too. Your own configs should go into `/etc/X11/xorg.conf.d/` and the `/etc/X11/xorg.conf` should remain untouched.

~/.xsession-errors

In case of any issues during the start of running of X, the errors will go here. So if you ran into an issue on your GUI startup, this is the file you should check to see what went wrong.

xhost

This command controls the access to the X server. If you are on a X server and run `xhost` it tells you the access status.

Copy

```
$ xhost
access control enabled, only authorized clients can connect
SI:localuser:jadi
```

As you can see only authorized clients can connect. To open it for all:

Copy

```
jadi@funlife:~$ xhost +
access control disabled, clients can connect from any host
```

And for closing it again:

Copy

```
jadi@funlife:~$ xhost -
access control enabled, only authorized clients can connect
```

Or open it for only one specific IP:

Copy

```
jadi@funlife:~$ xhost +192.168.42.85
192.168.42.85 being added to access control list
jadi@funlife:~$ xhost
access control enabled, only authorized clients can connect
INET:192.168.42.85    (no nameserver response within 5 seconds)
SI:localuser:jadi
```

Now the `192.168.42.85` machine (REMOTE) can send its graphical requests to this machine. whats the usage? continue reading the next section (assuming this machine is called `192.168.42.80` (SERVER) and we opened the access from to its X11 for `192.168.42.85` (REMOTE)).

DISPLAY

This variable tells graphical programs where to send their graphical output. This is the default:

Copy

```
$ echo $DISPLAY
:0
```

So if I run any graphical program on my machine, its output will be displayed on the same machine.

But lets change it to the `192.168.42.80` machine (SERVER).. if you remember we just told it that this machine (REMOTE) can connect to it.

Copy

```
$ export DISPLAY=192.168.42.80:0
$ xeyes # the eyes will be shown on 192.168.42.80 machine
```

Cool? yes. But this might not wonk on your setup. The X does not listen for remote connection in most distros because of security concerns.

> In the next chapter, we will see how you can properly run graphical programs on remote servers and receive the GUI part on your side

xauth

The `xauth` program is used to edit and display the authorization information used in connecting to the X server. On xhost you see how you can open your X to remote IPs, using `xauth` you can do the same based on a shared "secret".

In this method, `X` creates an `.XAuthority` file in your home and whoever knows about the contents, can contact X.

> If your `X` is not working, one step might be `rm`ing the `.XAuthority` file and restarting the X.

Wayland

X11 is super old and is only usable via lots of patches and hacks. Thats why one of its developers started a modern display manager called *Wayland*. In recent years many distros are switching from the default *X11* to *Wayland* and keeping X11 as a fail-safe option. Wayland is more secure and much easier to maintain and I can assure you that in a couple of years we will see more and more of it.

106.2 Graphical Desktops

Weight: 1

Candidates should be aware of major Linux desktops. Furthermore, candidates should be aware of protocols used to access remote desktop sessions.

Key Knowledge Areas

- Awareness of major desktop environments
- Awareness of protocols to access remote desktop sessions

Terms and Utilities

- KDE
- Gnome
- Xfce
- X11
- XDMCP
- VNC
- Spice
- RDP

Desktop Environments

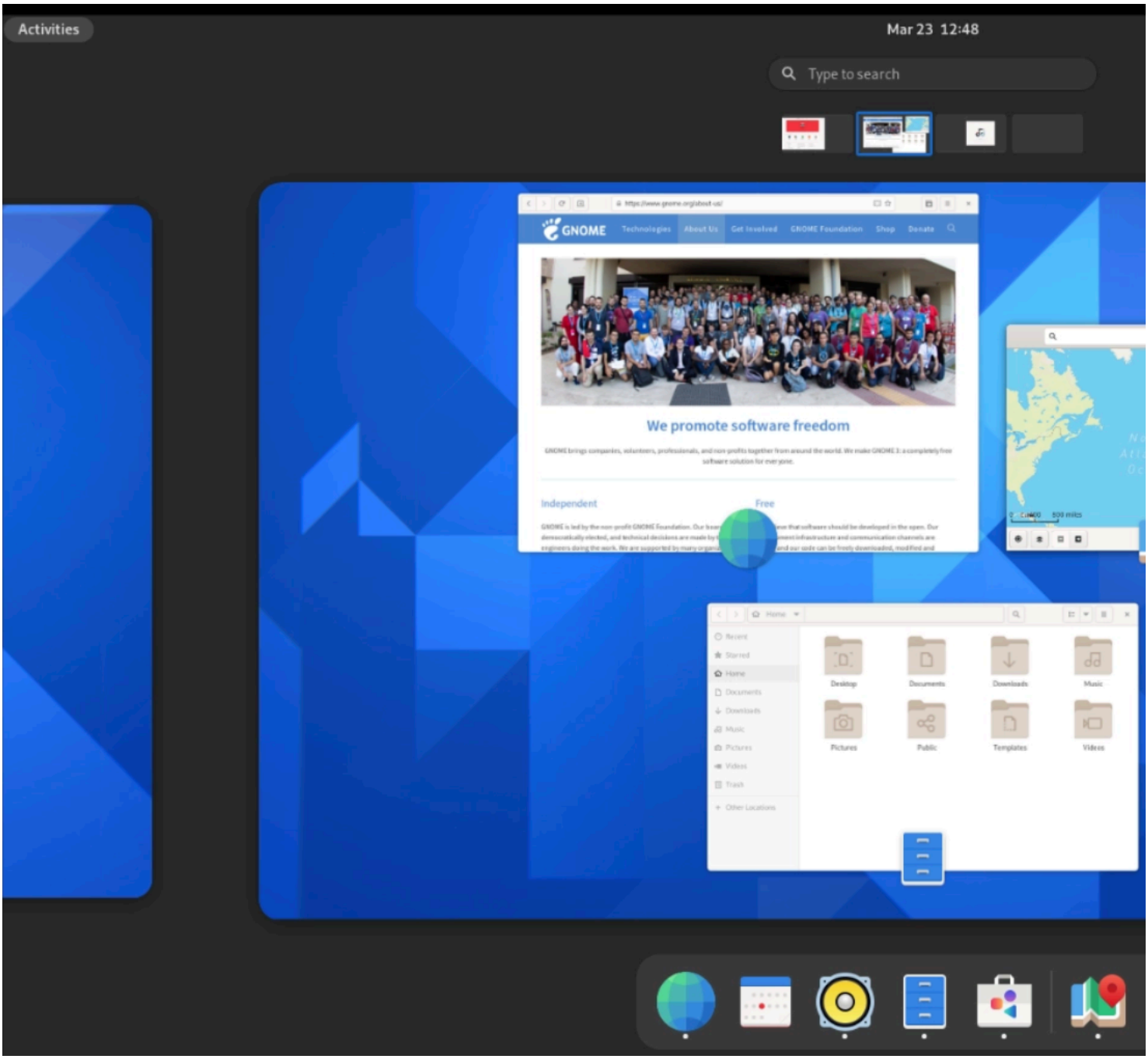
When your system boots in the graphical mode, you will see a graphical login page. This is based on the **X Display Manager Control Protocol (XDMCP)** which is responsible for the underlying login system. On top it different programs like GDM (for GNOME), SSDM (KDE) and a more general option XDM handle the theming and other aspects of login.

After the successful login, you will be in your GUI. Generally this is a **large** software package which manages your windows, mouse and has clickable icons and folder with drag and drop capability. There might even be a trashcan somewhere and lots of other softwares like calculators and an email client. These are called **Desktop Environments**. On the other hand, some people prefer to have a minimal window manager without all the bells and whistles and chose and add them based on their own preferences.

> A window manager is responsible for showing and managing windows on your GUI. Even the most minimalist GUI, needs a window manager to be function. But add more and more libraries and software to your window manager (say a calculator, calendar, email client, programming libs for that specific environment, maps, ...) and you will end up with what is called a Desktop Environment.

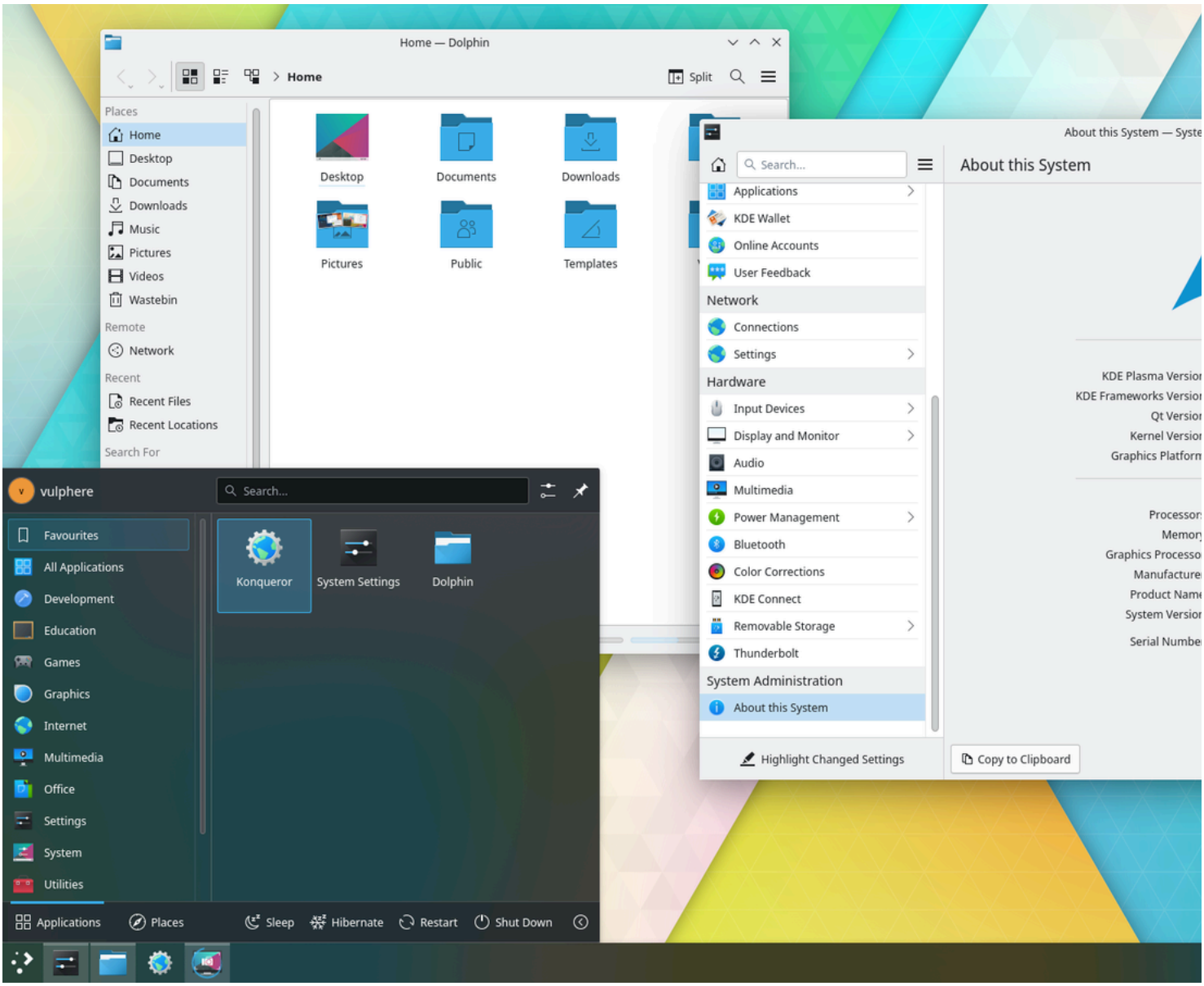
In this section we will review 3 of the most famous Desktop Environments and then will talk about "remote access" to desktop envs.

GNOME



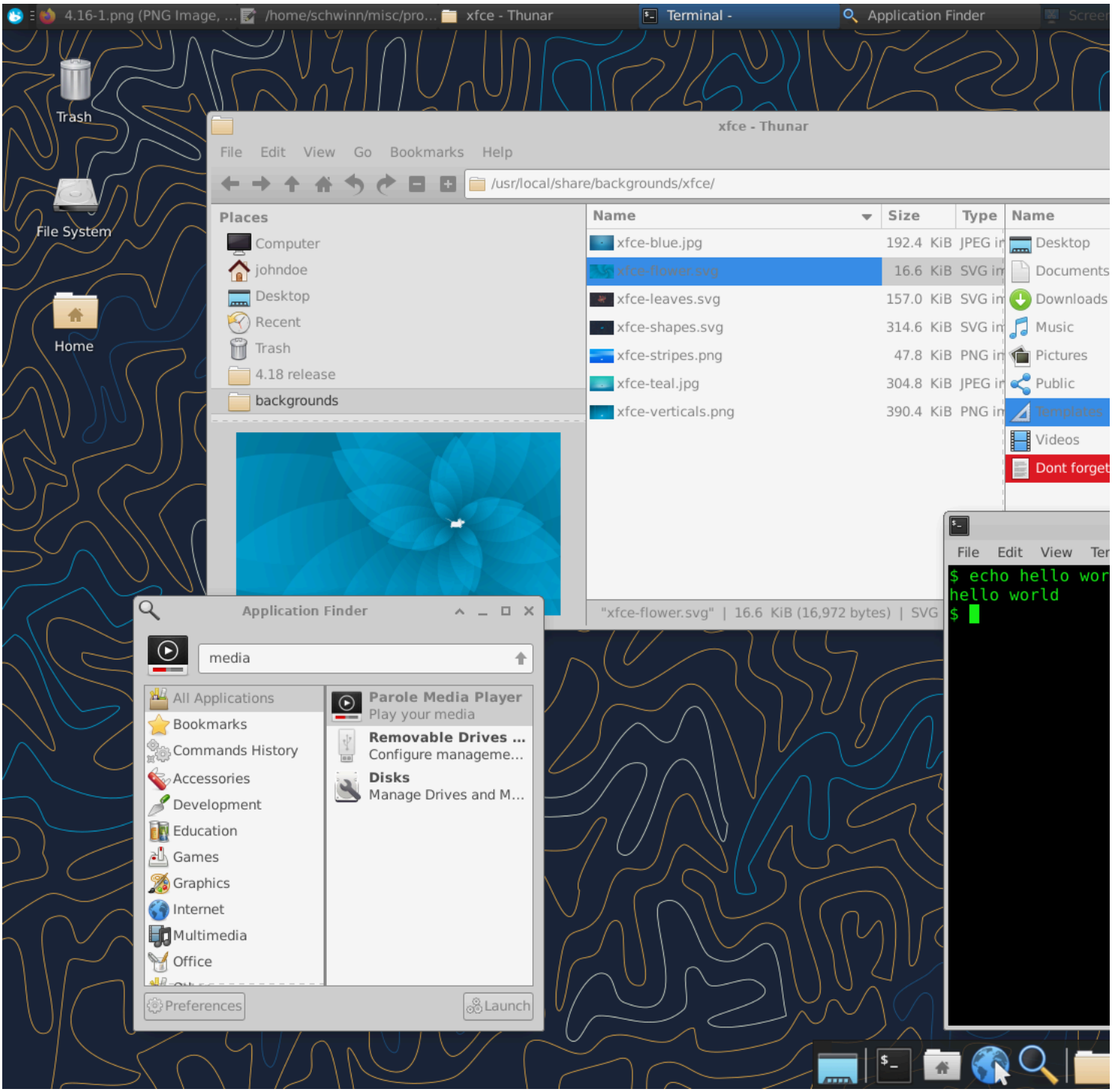
- Started at 1999
- Focuses on productivity & Accessibility
- Used by most GNU/Linux distros

KDE [Plasma]



- Started at 1996
- From version 5 (2016) re-branded to KDE Plasma
- Highly customizable
- Used during the Mars mission in NASA
- CERN uses KDE

XFCE



- Started at 1999
- Lightweight but beautiful
- Modular

Remote Connection to GUIs

There are many situation where you want to connect remotely to your GUI based system. This can be remote assistance to a friend, checking the status on your computer from your phone, use a GUI based program on a much stronger remote machine or as a colleague did, hide a laptop in the LAB and connect to it remotely and work from home when you should have been to the office (don't!).

Here we will do a quick review on some of these methods.

X Forwarding

This is not mentioned in this section in LPIC1 but it is super useful to know about. If you have an SSH server, you can connect to it and run GUI softwares there and get the GUI part of the program on your own machine. This is called X-Forwarding in ssh.

First make sure that the `X11Forwarding yes` is configured in `/etc/ssh/sshd_config` (and [restart the service is needed]). Then do your ssh as follow:

Copy

```
$ ssh -X server_ip.address.net
$ xeyes
```

And they eyes will appear on your own machine! You can do the same to run a browser or a cpu hungry simulation software.

VNC

Virtual Network Computer (VNC) started long time ago and is still active and alive. It is multi platform and uses RFB protocol. Its default port is 5900 + [display number, usually is 1 by default] so you need to use 5901 port to connect via VNC.

The good point about VNC is its flexibility and clients on all platform but it lacks security.

Spice

The **Simple Protocol for Independent Computing Environment** started as a closed source software but become open source when RedHat bought the company in 2008. It can be used to connect to KVM virtual machines (a very common virtual machine system) and has some advantages like fast speeds similar to local connections and low CPU usage. There is one server implementation but you can find many clients, including GNOMEs boxes program.

RDP

Using softwares like `Xrdp` you can start listening for Remote Desktop Protocol (RDP) connections on your machine on port 3389 by default. The traffic is encrypted by default and there are lots of free and open RDP clients available.

106.3 Accessibility

Weight: 1

Demonstrate knowledge and awareness of accessibility technologies.

Key Knowledge Areas

- Basic knowledge of visual settings and themes.
- Basic knowledge of assistive technology.

Terms and Utilities

- High Contrast/Large Print Desktop Themes
- Screen Reader
- Braille Display
- Screen Magnifier
- On-Screen Keyboard
- Sticky/Repeat keys
- Slow/Bounce/Toggle keys
- Mouse keys
- Gestures
- Voice recognition

Linux is for everyone

Some people have physical complications. Some can not see well, some can not see at all and some can not user their finger as most people do. Linux have 3 answers:

1- AccessX helps people with physical problems to use keyboard/mouse
2- Visual Settings help people with vision problems by magnifying the screen and things like that
3- Assistive Technologies are things like text-to-speech (tts) and reads the screen for people with visual problem

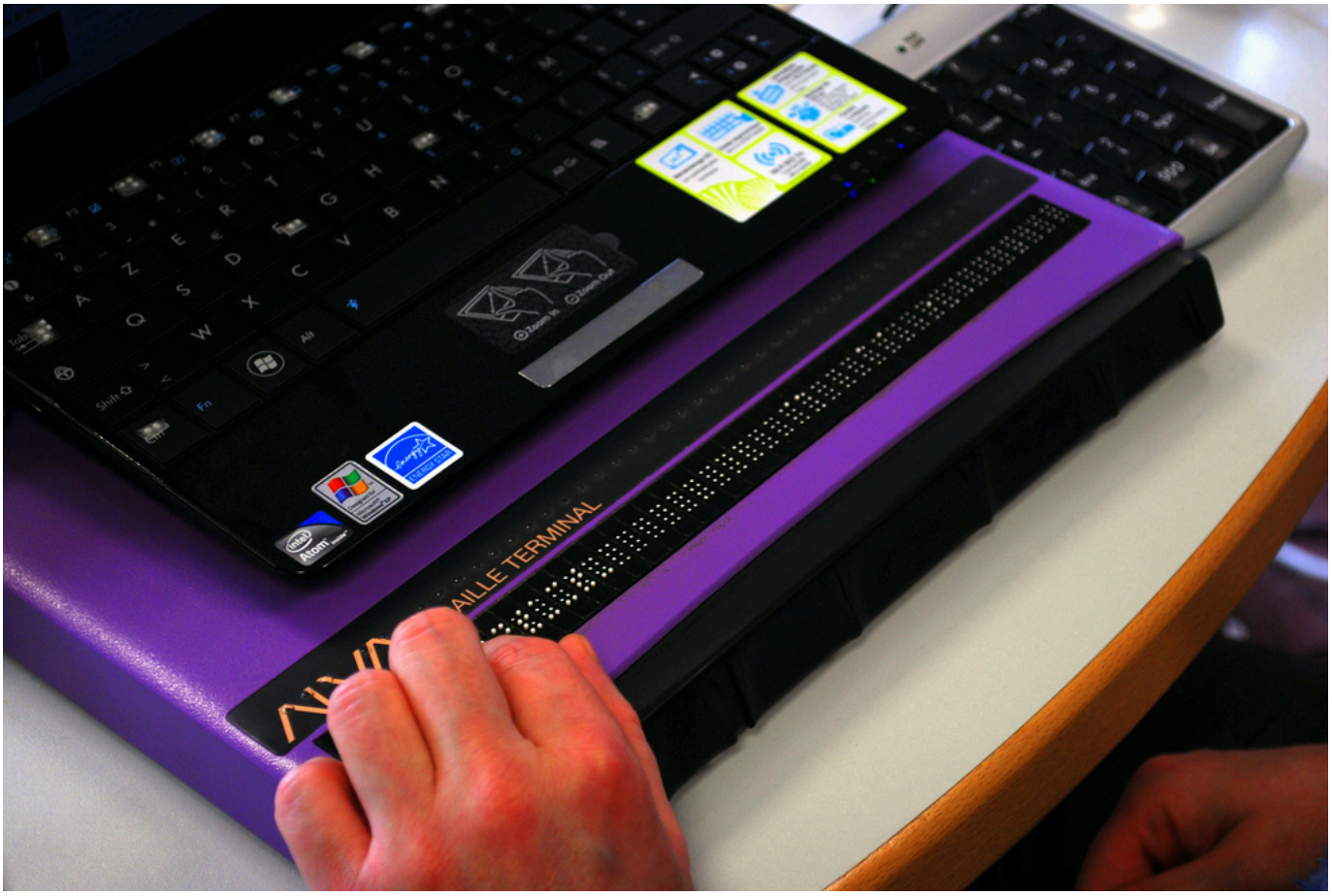
These options are available in display managers (login screen) and in major desktops (like GNOME, KDE Plasma, xfce, ...).

In Gnome the config is located at Settings ~ Universal Access. The configurations are categorized and are as follows:

- High Contrast
- Zoom
- Large text
- Screen Reader
- Screen Keyboard (show a keyboard on screen)
- Visual Alerts (instead Beeps, flash the screen)
- Sticky Keys (Press shift, then press a -> capital A)
- Slow Keys (do not repeat keys after pressing a key for few seconds)
- Bounce Keys (if you hit a key twice fast, it won't accept the second one)
- Mouse Keys (Arrow keys on number path will work as a mouse)
- Simulate Secondary Click (by holding down the click)
- Hover click (click by waiting on a button)

TTS

Applications like **Orca** or **Emacspeak** can read the dialog boxes to you so you can decide what the answer only by hearing.



Above you can see a Braille Display. This device will output text as braille for visually impaired people, although using a modern Text To Speech or screen reader might be more practical when possible.

Voice recognition

After the recent advancements in Voice recognition, this is one of the best methods for people with disabilities to control their OSs.

107.1 Manage user and group accounts and related system files

Weight: 5

Candidates should be able to add, remove, suspend and change user accounts.

Key Knowledge Areas

- Add, modify and remove users and groups.
- Manage user/group info in password/group databases.
- Create and manage special purpose and limited accounts.

Terms and Utilities

- `/etc/passwd`
- `/etc/shadow`
- `/etc/group`
- `/etc/skel/`
- `chage`
- `getent`
- `groupadd`
- `groupdel`
- `groupmod`

- `passwd`
- `useradd`
- `userdel`
- `usermod`

Passwords

Changing Passwords

Each user can change her password using the `passwd` command:

Copy

```
$ passwd
Changing password for jadi.
(current) UNIX password:
New password:
Retype new password:
passwd: password updated successfully
```

If the password is too short or too similar to the previous one or even a dictionary word, or equal to the username and such, the `passwd` command may refuse to change it. Also note that the `passwd` commands asks for the *current password* first to make sure that it is being changed by yourself.

> The root user can change any users password to anything (weak passwords) without providing their current password:

Copy

```
# passwd jadi
New password:
BAD PASSWORD: it does not contain enough DIFFERENT characters
BAD PASSWORD: is too simple
Retype new password:
passwd: password updated successfully
```

Users and groups

Linux is a multi-user system and managing these users is part of system admin's job. As the first step, you should be able to **add**, **remove** and **modify** users.

Linux also has the concept of **groups**. You can define groups, give privileges to them and add users to these groups. As you've already seen in the `chmod` command, you might give access the permission to read from `/dev/cdrom` to a group called `cdrom` and then add whoever needs to read from the CD-ROM to this group.

Please note that each user can be a member of many groups but only one of these will be her *Primary* group. In contrast, each file can be owned only by one group.

Managing Users

Adding users

To add a new user to your system, use the `useradd` command. These are the main switches:

switch	meaning
-d	home directory (-d /home/user)
-m	create home directory
-s	specify shell
-G	add to additional groups
-c	comment. most of the time, users actual name. Use quotes if comments has spaces or special characters in them

On some systems `useradd` creates the home directory and on some, you have to specify the `-m` switch yourself. It is a good practice to use it all the time.

When a new user directory is being created, the system will copy the contents of `/etc/skel` to their home dir. `/etc/skel` is used as a template for the home of users.

> Another command to add users is `adduser`. It will ask for the password, home directory, etc.. and will create the user.

Modifying users

To modify a user, you should use the `usermod` command. It supports most of the `useradd` switches. For example you can change *jadi*'s login shell by issuing `usermod -s /bin/csh jadi`. But there are 3 more switches too:

switch	meaning
-L	lock this account
-U	Unlock the account
-aG	append to more groups (say `usermod -aG wheel jadi`)

> Note: If you do `usermod -G wheel,users jadi`, jadi will be the member of these two groups ONLY. That is why we use `-aG newgoup` to *append to groups* to add this group to jadi's groups. `-G` is like saying "jadi's groups are ..." and `-aG` is like "add jadi to these groups too".

Deleting users

If you want to remove a user, use `userdel`. Straight forward:

Copy

```
userdel jadi
```

If you add the `-r` switch, the home directory and mail spool will be erased too!

Managing Groups

It is kind of same as users, you can do `groupadd`, `groupdel` and `groupmod`. Each group has an *id* and a *name*.

Copy

```
# groupadd -g 1200 newgroup
```

adds a group called *newgroup* with id 1200. If needed, the root user can change a groups ID (to 2000) by issuing `groupmod -g 2000 newgroup` or deleting the group by `groupdel newgroup`.

Obviously if root deletes a group with members, people wont be deleted! They will just wont be the members of that group anymore.

> In most systems, the user id or group id's of the users and groups created by the admin will have the id of 1000, 1001, 1002, ...

Files related to users and groups

/etc/passwd

This file contains all the user information on your system:

Copy

```
tail /etc/passwd
scard:x:491:489:Smart Card Reader:/var/run/pcscd:/usr/sbin/nologin
sshd:x:493:491:SSH daemon:/var/lib/sshd:/bin/false
statd:x:488:65534:NFS statd daemon:/var/lib/nfs:/sbin/nologin
tftp:x:496:493:TFTP account:/srv/tftpboot:/bin/false
lightdm:x:10:14:Light Display Manager:/var/lib/lightdm:/bin/false
wwwrun:x:30:8:WWW daemon apache:/var/lib/wwwrun:/bin/false
jadi:x:1000:100:jadi:/home/jadi:/bin/bash
svn:x:485:482:user for Apache Subversion svnserve:/srv/svn:/sbin/nologin
privoxy:x:484:480:Daemon user for privoxy:/var/lib/privoxy:/bin/false
```

As you can see the format is:

Copy

```
username:x:userid:primary group id:Name and comments:home dir:shell
```

As you can see, the 2nd field is shown as "`x`" here. In the old days the password or the hashed password of the user was shown in this file but nowadays that is moved to the `/etc/shadow` file.

> Note: /etc/passwd should be readable to all users and this makes it a very bad place to keep users password! Thats why we see a `x` instead of the password, this indicates that the main passwords should be looked up from `/etc/shadow` file.

Note how *special users* like `lightdm` have `/bin/false` as their shell; this prevents them from logging into the system for real. In old days hackers used to try these accounts to login.

/etc/shadow

This file contains password (hashed passwords) of the users. See how the `/etc/passwd` is readable for all but `/etc/shadow` is only readable for root and members of the `shadow` group:

Copy

```
# ls -ltrh /etc/passwd /etc/shadow
-rw-r--r-- 1 root root   1.9K Oct 28 15:47 /etc/passwd
-rw-r----- 1 root shadow  851 Oct 29 19:06 /etc/shadow
```

But what is in it?

Copy

```
# tail /etc/shadow
scard:!:16369::::::
sshd:!:16369::::::
statd:!:16369::::::
tftp:!:16369::::::
uucp:*:16369::::::
lightdm:*:16369::::::
jadi:$6$enk5I3bv$uSQrRpen7m9xDapYLgwgh3P/71OLZUgj31n8AwzgIM2lA5Hc/BmRVAMC0eswdBGkseuXSvmaz0lsYFtduvuqUo:16737:0:99999:7:::
svn:!:16736::::::
privoxy:!:19473::::::
```

Wow! Jadi has an encrypted password there. Some numbers are following that encrypted password too: **`16737:0:99999:7:::`**. What do the mean? The following table tells you.

filed	meaning
19473	When was the last time this password changes
0	User wont be able to change the password 0 days after each change
99999	After this many days, the user HAVE to change his password
7	...and the user will be informed 7 days before the expiration to change his password

> Note: there numbers are "days after 1st of January 1970" or the Epoch time in days. For example 16737 means 16737 days after 1st Jan 1970. Strange but practical!

But we do not need to change these strange number manually. If needed, we can use the `chage` tool to change these numbers. If you issue the `chage jadi` the system will prompt you for all the parameters one by one. Also it is possible to use switches to change specific parameters on command line.

switch	meaning
-l	list information
-E	Set the expiration date. Date can be a number, in YYYY-MM-DD format or -1 which will mean *never*

Copy

```
# chage -l jadi
Last password change                    : Apr 26, 2023
Password expires                      : never
Password inactive                      : never
Account expires                         : never
Minimum number of days between password change        : 0
Maximum number of days between password change        : 99999
Number of days of warning before password expires    : 7
```

> Note: `!` means the account is locked and the user can not login into the system using the account. On some systems (like RedHat) you may also see `!!` which means the account has never been used.

`/etc/group`

This file contains the groups and their IDs.

Copy

```
# tail /etc/group
avahi:x:486:
kdm:!:485:
mysql:x:484:
winbind:x:483:
at:x:25:
svn:x:482:
vboxusers:x:481:
input:x:1000:jadi,joe
privoxy:x:480:
```

> Note: See that `x` there? Theoretically groups can have passwords but it is never used in any distro! The file is `/etc/gshadow`

checking user info

Previously you saw the `chage -l jadi` but there are more commands for checking user status. One is `id`:

Copy

```
# id jadi
uid=1000(jadi) gid=100(users) groups=1000(input),100(users)
```

Another solution is `getent` (for **get entry**). It can query important *databases* for specific entries. These databases include `/etc/passwd`, `/etc/hosts`, `/etc/shadow`, `/etc/group`, **...**

Copy

```
funlife:~ # getent group tor
tor:x:479:
funlife:~ # getent passwd jadi
jadi:x:1000:100:jadi:/home/jadi:/bin/bash
funlife:~ # getent shadow jadi
jadi:$6$enk5I3bv$uSQrRpen7m9xDapYLgwgh3P/71OLZUgj31n8AwzgIM2lA5Hc/BmRVAMC0eswdBGkseuXSvmaz0lsYFtduvuqUo:16737:0:99999:7:::
```

107.2 Automate system administration tasks by scheduling jobs

Weight: 4

Candidates should be able to use cron and systemd timers to run jobs at regular intervals and to use at to run jobs at a specific time.

Key Knowledge Areas

- Manage cron and at jobs.
- Configure user access to cron and at services.
- Understand systemd timer units.

Terms and Utilities

- `/etc/cron.{d,daily,hourly,monthly,weekly}/`
- `/etc/at.deny`
- `/etc/at.allow`
- `/etc/crontab`
- `/etc/cron.allow`
- `/etc/cron.deny`
- `/var/spool/cron/`
- `crontab`
- `at`
- `atq`
- `atrm`
- systemctl
- systemd-run

There are many cases where we need to run some commands at specific times or intervals. For example you may want to run your backup process every morning at 03:00 or refresh the list of files on your web index on hourly basis.

There are different methods to do so in the Linux world. The most commons are the `crontab` to run tasks on specific intervals, `at` to run a commands on specific time and also utilities provided by `systemd`.

Crontab format

Crontab works with a configuration file which tells it when to run a command. For example you can say "run my commands when the hour is 03 and the minute is 00" or "run my command when its Saturday and Hour is 10 and minute is 00" or "Run whenever the minute is 00".

Each crontab configuration line has 6 fields. The first 5 specify a specific interval and whatever after than is used as the "command".

Copy

```
A    B    C    D    E    command and arguments
```

filed	Meaning	values
A	minute	0-59
B	hour	0-23

filed	Meaning	values
C	day of month	1-31
D	month	1-12 (or names, see below)
E	day of week	0-7 (0 or 7 is Sunday, or use names)

If you replace a field with `*` it means "whatever" or "all" or "any". Also if you have `@reboot` or `@daily` instead of time fields, the command will be run once after the reboot or daily. Lets see some examples:

Copy

```
5 0 * * *       $HOME/bin/daily.job >> $HOME/tmp/out 2>&1

# run at 2:15pm on the first of every month -- output mailed to paul
15 14 1 * *     $HOME/bin/monthly

# run at 10 pm on weekdays, annoy Joe
0 22 * * 1-5    mail -s "It's 10pm" joe%Joe,%%Where are your kids?%

23 0-23/2 * * * echo "run 23 minutes after midn, 2am, 4am ..., everyday"
5 4 * * sun     echo "run at 5 after 4 every sunday"

*/5 * * * *    echo "each 5 mintues"

42 8,18 * * 1-5    echo "8:42 and 18:42 and only on weekdays (monday till friday)"

@reboot        echo "runs after the reboot"
```

As you can see in above examples, `/` can be used to slice the time (so `*/5` in minutes means every five minutes) or use the `,` to indicate more than one number (so `8,18` means 8 and 18).

> When using `*` on the first filed, you are running your command on every minute.
>
> Something like `42 8 1 1 0` runs ONLY when the 1st of Jan is a Sunday!

When a cron runs, the output will be emailed to the owner of the cron.

user specific crons

Cron is a linux service. To see your crons you can use `crontab -l` (list) and for editing them you can use `crontab -e` (edit) which will open the cron files with a special editor and will validate and then load your crons after you are finished.

The files will be saved at `/var/spool/cron/tabs/` or `/var/spool/crontabs`:

Copy

```
# cat /var/spool/cron/tabs/jadi
# DO NOT EDIT THIS FILE - edit the master and reinstall.
# (/tmp/crontab.khObLu installed on Thu Oct 29 22:04:43 2023)
# (Cronie version 4.2)
# Example of job definition:
# .---------------- minute (0 - 59)
# |  .------------- hour (0 - 23)
# |  |  .---------- day of month (1 - 31)
# |  |  |  .------- month (1 - 12) OR jan,feb,mar,apr ...
# |  |  |  |  .---- day of week (0 - 6) (Sunday=0 or 7) OR sun,mon,tue,wed,thu,fri,sat
# |  |  |  |  |
# *  *  *  *  * command to be executed
* * * * * date >> /tmp/date.cron.txt
```

> never edit these files directly; Use `crontab -e` instead.

system wide cron

There is file called `/etc/crontab`. This looks like a normal user file opened with `crontab -e` but has one extra filed:

Copy

```
A    B    C    D    E    USER    command and arguments
```

This file should be edited with an editor directly and we can mention which user runs this commands.

Copy

```
# cat /etc/crontab
SHELL=/bin/sh
PATH=/usr/bin:/usr/sbin:/sbin:/bin:/usr/lib/news/bin
MAILTO=root
```

```
#
# check scripts in cron.hourly, cron.daily, cron.weekly, and cron.monthly
#
-*/15 * * * *   root  test -x /usr/lib/cron/run-crons && /usr/lib/cron/run-crons >/dev/null 2>&1
```

> Note: Have a look at the first three lines. It configures the shell which will run the commands and the PATH variable plus who will get the output emails.

As you can see this default crontab runs other crons! These are hourly, daily, weekly and monthly crons. Lets have a look at them in the next section.

System hourly, daliy, weekly, monthly, .. crons

We have some system level crontab files in `/etc/cron.d/` too. They look like general crontab files. For example this is one that `munin` program creates:

Copy

```
$ cat /etc/cron.d/munin
#
# cron-jobs for munin
#

MAILTO=root

*/5 * * * *     munin if [ -x /usr/bin/munin-cron ]; then /usr/bin/munin-cron; fi
14 10 * * *     munin if [ -x /usr/share/munin/munin-limits ]; then /usr/share/munin/munin-limits --force --contact nagios --contact old-nagi
19:12:37 jadi@ocean:~$ cat /etc/cron.weekly/man-db
```

But there are also directories called `cron.hourly`, `cron.daily`, `cron.weekly` & `cron.monthly`. As their name indicates, whatever script / file you put in them, will be run hourly, daily, weekly & monthly. This makes general cleanup, backup, ... tasks much cleaner. Its enough to copy your script in any of these directories and let them run on these intervals.

Copy

```
$ sudo tree /etc/cron*
/etc/cron.d
├── munin
├── munin-node
├── php5
└── sendmail
/etc/cron.daily
├── apache2
├── apport
├── apt
├── aptitude
├── bsdmainutils
├── dpkg
├── logrotate
├── man-db
├── mlocate
├── passwd
├── popularity-contest
├── sendmail
├── update-notifier-common
└── upstart
/etc/cron.hourly
/etc/cron.monthly
/etc/crontab [error opening dir]
/etc/cron.weekly
├── apt-xapian-index
├── fstrim
├── man-db
└── update-notifier-common
```

Lets have a look at one of the cron.d files:

Copy

```
$ cat /etc/cron.daily/mlocate
#! /bin/bash

set -e

[ -x /usr/bin/updatedb.mlocate ] || exit 0

if which on_ac_power >/dev/null 2>&1; then
    ON_BATTERY=0
    on_ac_power >/dev/null 2>&1 || ON_BATTERY=$?
```

```
    if [ "$ON_BATTERY" -eq 1 ]; then
    exit 0
    fi
fi

# See ionice(1)
if [ -x /usr/bin/ionice ] &&
    /usr/bin/ionice -c3 true 2>/dev/null; then
    IONICE="/usr/bin/ionice -c3"
fi

flock --nonblock /run/mlocate.daily.lock $IONICE /usr/bin/updatedb.mlocate
```

at

We say that crontab runs commands on specific intervals but what will happen if you needed to run a command only once? Here `at` is your friend.

Copy

```
$ at now + 1 min
warning: commands will be executed using /bin/sh
at> touch /tmp/at
at> <EOT>
job 3 at Thu Oct 29 22:12:00 2015
```

> Note: As always, at the end of input we press Ctrl+D

If you want to check what is in the queue you can use `atq` and then try `atrm 4` to delete job number 4;

Copy

```
$ at teatime
warning: commands will be executed using /bin/sh
at> echo "tea time is 4pm"
at> <EOT>
job 4 at Fri Oct 30 16:00:00 2015
jadi@funlife:~$ at tomorrow
warning: commands will be executed using /bin/sh
at> echo "tomorrow this time"
at> <EOT>
job 5 at Fri Oct 30 22:15:00 2015
jadi@funlife:~$ at 17:30
warning: commands will be executed using /bin/sh
at> echo "this is the first 17:30"
at> <EOT>
job 6 at Fri Oct 30 17:30:00 2015
jadi@funlife:~$ atq
5    Fri Oct 30 22:15:00 2015 a jadi
4    Fri Oct 30 16:00:00 2015 a jadi
6    Fri Oct 30 17:30:00 2015 a jadi
jadi@funlife:~$ atrm 4
jadi@funlife:~$ atq
5    Fri Oct 30 22:15:00 2015 a jadi
6    Fri Oct 30 17:30:00 2015 a jadi
```

managing cron access

To control the access to the cron system, we have 4 files:

Copy

```
/etc/cron.allow
/etc/cron.deny

/etc/at.allow
/etc/at.deny
```

In most systems none of these files exist but if you create them, they will become active as follow:

file extension	functionality
.allow	ONLY users mentioned in this file are allowed to run this service. All other users will be denied
.deny	Everybody can use the service except the users mentioned in this file

systemd & systemd-run

Nowadays most GNU/Linux distributions use `systemd` which has its own **timers** as an alternative to cron to schedule your tasks. Timer unit files use the `.timer` suffix, and for each of these there must be a corresponding unit file which describes the unit to be activated when the timer

activates. By default, a timer activates a service with the same name, except for the suffix.

Timers have a [Timer] section that specifies when scheduled jobs should run. There are two types of timers. One is the **readl-time timer** which works with `OnCalendar=` option and defines kind of close to cron timers (based on calendar event expressions). This is the syntax:

Copy

```
OnCalendar=DayOfWeek Year-Month-Day Hour:Minute:Second
```

DayOfWeek is optional and you can use a 3 letter abbreviation (say Mon)

The `*`, `/` and `,` work the same in systemd and cron configurations but in systemd you can also use `..` between two values to indicate a contiguous range.

For example, to run the service named `/etc/systemd/system/myservice.service` at 02:42 on the first Saturday of each month, this config line should be added to the `/etc/systemd/system/myservice.timer` unit file.

Copy

```
[Unit]
Description=Run the blah blah service

[Timer]
OnCalendar=Sat *-*-1..7 02:42:00
Persistent=true

[Install]
WantedBy=timers.target
```

It is also possible to define **monotonic** timers. They activate after some time has elapsed from a specific start point (say when the machine was booted up or when the timer itself is activated). You can chose between any of the following:

config	meaning
OnActiveSec	time after unit activation
OnBootSec	time after system boot
OnStartupSec	time after the service manager is started. For system services, this is almost equal to OnActiveSec. Use this for user services where the service manager is started at user login.
OnUnitActiveSec	time after the corresponding service was last activated
OnUnitInactiveSec	time after the corresponding service was last deactivated

And you can use any of usec, msec, seconds, minutes, hours, days, weeks, months, years in front. So for example the

Copy

```
[Timer]
OnUnitActiveSec=5minutes 20seconds
```

will activate the corresponding unit 5min and 20secs after the last activation of this unit service.

Please note that after adding any timer, you need to enable (and probably start) it:

Copy

```
# systemctl enable myservice.timer
# systemctl start myservice.timer
```

And also do not forget to do a `systemctl daemon-reload` after each change.

To check all the scheduled jobs, run the `systemctl list-timers` command. This will show you all the active timers. Use the `--all` switch to check all.

There are few shortcuts for times too. You can use these instead of the cron format to set timers:

shortcut	when it runs
hourly	beginning of each hour
daily	One a day at midnight
weekly	midnight on Monday
monthly	midnight on the 1st of each month
yearly	midnight on the 1st of Jan

This is one example from a recent Fedora installation:

Copy

```
# cat system/timers.target.wants/plocate-updatedb.timer
[Unit]
Description=Update the plocate database daily

[Timer]
OnCalendar=daily
RandomizedDelaySec=12h
AccuracySec=20min
Persistent=true

[Install]
WantedBy=timers.target
```

If you want to check the output of your timers, you can use the `journalctl`.

> as any other `systemd` activity, use the `--user` switch to use it on your user level and not systemwide.

At the end, you should know about the `systemd-run` command. It can start a transient service or scope and run specific commands in it.

So a command like:

Copy

```
systemd-run --on-active="2hours" --unit="helloworld.service"
```

Will run the helloworld.service every 2 hours or this:

Copy

```
sudo systemd-run --on-active="2hours" /usr/local/bin/helloworld.sh --language=en_US
```

will run that specific script with shown parameters every 2 hours.

Since this is new and there are not much sources about it, refer to Suse docs to learn more.

107.3 Localisation and internationalisation

Weight: 3

Candidates should be able to localise a system in a different language than English. As well, an understanding of why `LANG=C` is useful when scripting.

Key Knowledge Areas

- Configure locale settings and environment variables.
- Configure timezone settings and environment variables.

Terms and Utilities

- `/etc/timezone`
- `/etc/localtime`
- `/usr/share/zoneinfo/`
- `LC_*`
- `LC_ALL`
- `LANG`
- `TZ`
- `/usr/bin/locale`
- `tzselect`
- `timedatectl`
- `date`
- `iconv`
- UTF-8
- ISO-8859
- ASCII
- Unicode

timezone

The earth is large and round so we have different ***timezones*** and different ***languages***! There are even talks about a time zone for Mars and if we encounter aliens, they might also have their own languages.

This makes a few problems: 1. we may prefer our local language for our Linux 2. we need to tell linux about our timezone 3. programs may ask linux about our preferences on languages, keyboard configuration, and .. 4. and even if you do not case about yourself, your machines needs your timezone to be able to run tasks on needed times

In this section I will show you the answers Linux has to overcome these issues. But lets start simple.

On linux systems you can use `date` and `cal` commands to check the date and the calendar. It is possible to print a custom date using `+` formatter:

Copy

```
[jadi@fedora ~]$ date
Fri Jun  2 02:07:35 PM EDT 2023
[jadi@fedora ~]$ date +'%Y%m%d-%M'
20230602-08
[jadi@fedora ~]$ date +'%Y%m%d-%H%M'
20230602-1408
```

to get more info you can try `timedatectl`:

Copy

```
[root@fedora ~]# timedatectl
               Local time: Fri 2023-06-02 14:23:34 EDT
           Universal time: Fri 2023-06-02 18:23:34 UTC
                 RTC time: Sat 2023-06-03 02:37:51
                Time zone: America/New_York (EDT, -0400)
System clock synchronized: no
              NTP service: active
          RTC in local TZ: no
```

Timezone determines what is your time difference comparing with a reference timezone. This way you can talk about times regardless of your location. In another words, I can tell you "start the change request at 02:30 UTC" and we both know when the change will be started in our own timezone (mine will be 1 hour ahead so I'll start at 3:30 my time).

Most distributions will ask you about your time zone during the installation. You can change this later, either by a GUI in the system settings or using a text based user interface (TUI) like the older `tzconfig` or a newer `tzselect`.

Copy

```
[jadi@fedora ~]$ tzselect
Please identify a location so that time zone rules can be set correctly.
Please select a continent, ocean, "coord", or "TZ".
1) Africa                                7) Europe
2) Americas                              8) Indian Ocean
3) Antarctica                            9) Pacific Ocean
4) Asia                                 10) coord - I want to use geographical coordinates.
5) Atlantic Ocean                       11) TZ - I want to specify the timezone using the Posix TZ format.
6) Australia
#? 5
Please select a country whose clocks agree with yours.
1) Bermuda                    4) Faroe Islands                 7) Spain
2) Cape Verde                     5) Portugal
```

```
3) Falkland Islands               6) South Georgia & the South Sandwich Islands
#? 1

The following information has been given:

    Bermuda

Therefore TZ='Atlantic/Bermuda' will be used.
Selected time is now:   Fri Jun  2 15:15:06 ADT 2023.
Universal Time is now:  Fri Jun  2 18:15:06 UTC 2023.
Is the above information OK?
1) Yes
2) No
#? 1

You can make this change permanent for yourself by appending the line
    TZ='Atlantic/Bermuda'; export TZ
to the file '.profile' in your home directory; then log out and log in again.

Here is that TZ value again, this time on standard output so that you
can use the /usr/bin/tzselect command in shell scripts:
Atlantic/Bermuda
```

When finished this program will suggest you to set a variable called `TZ` as follow to set *your own* time zone, but not the systems:

Copy

```
TZ='Atlantic/Bermuda'; export TZ
```

or use it once to check time in a specific city:

Copy

```
$ env TZ='Asia/Tokyo' date
Sat Jun  3 03:25:40 AM JST 2023
```

Configuring timezone

The `/usr/share/zoneinfo/` directory contains all the timezone info. These are binary files. If you need to change your system-wide timezone you need to short link the `/etc/localtime` to one of these files:

Copy

```
# ls -ltrh /etc/localtime
lrwxrwxrwx. 1 root root 38 Apr 26 09:41 /etc/localtime -> ../usr/share/zoneinfo/America/New_York
```

This file should be replaced by the correct file from `/usr/share/zoneinfo/`. It is nicer to make a symbolic link rather than copying the actual file. This will prevent the issues during next upgrades.

Configuring Languages

To check the status of current selected system language, use the `locale` command:

Copy

```
[root@fedora ~]# locale
LANG=en_US.UTF-8
LC_CTYPE="en_US.UTF-8"
LC_NUMERIC="en_US.UTF-8"
LC_TIME="en_US.UTF-8"
LC_COLLATE="en_US.UTF-8"
LC_MONETARY="en_US.UTF-8"
LC_MESSAGES="en_US.UTF-8"
LC_PAPER="en_US.UTF-8"
LC_NAME="en_US.UTF-8"
LC_ADDRESS="en_US.UTF-8"
LC_TELEPHONE="en_US.UTF-8"
LC_MEASUREMENT="en_US.UTF-8"
LC_IDENTIFICATION="en_US.UTF-8"
LC_ALL=
```

These are environment variables telling system what languages to use. Here I'm using `LANG=en_US.UTF-8` which means I'm using English with US variant and UTF-8 encoding.

> UTF-8 and other encodings will be discussed a bit later in this chapter

Other variables you see above, tell the system how to show different things based on localisation systems. For example if we change the `LC_TIME` to `"en\_GB.UTF-8"` the time will be printed in Great Britain format.

Another important settings is `LC_ALL`. It can be used to change **ALL** settings. If you do a `export LC_ALL=fa_IR.UTF-8`, all the settings will be set to that one, with no exception. It is always possible to `unset LC_ALL`.

LANG=C

The `LANG=C` settings is used in many programs and scripts to make sure that the system will create coherent and predictable results in all environments. For example to make sure that the sort order will be in Binary order and all language settings will be default (en.US).

changing or adding locales

This is not a part of LPIC exam but it is good to know that on a debian based machine, you can change, add or set your default *locales* using `dpkg-reconfigure locales`.

There is also this file:

Copy

```
[root@fedora ~]# cat /etc/locale.conf
LANG="en_US.UTF-8"
```

and you can also add your desired `LANG` config to your `~/.bash_profile` or `~/.profile`.

> on `systemd` use `localectl` to check locale or use `localectl set-locale LANG=en_US.UTF-8` to set it.

Character Encoding

ACSII

Computers used to work with 7bit characters encoding. That would give us 128 characters which was enough for numbers, punctuation and digits!

ISO-8859

It had more characters and a lots of sets for Thai, Arabic and other languages but still had ASCII character sets.

UTF-8

The `Unicode Transformation Format` is the newest encoding method. It is a real universal encoding with characters not only for all written languages but also for fun characters like ¾, ♠, π and ⚤. It is backward compatible with the ASCII and uses 8 bit code units (**not 8 bit coding!**). In most cases it is a good idea to use UTF-8 and be sure that your system will work in practically all cases.

iconv

If you needed to convert coding to each other, the command is `iconv`. The `-l` switch will show you all the available codings:

Copy

```
iconv -f WINDOWS-1258 -t UTF-8 /tmp/myfile.txt
```

> Note: -f is for "from" and -t is for "to". Easy to remember

In 2023 you will seldom need this command but it is a must to know it, specially if you are living in a non US country!

108.1 Maintain system time

Weight: 3

Candidates should be able to properly maintain the system time and synchronize the clock via NTP.

Key Knowledge Areas

- Set the system date and time.
- Set the hardware clock to the correct time in UTC.
- Configure the correct timezone.
- Basic NTP configuration using `ntpd` and `chrony`.
- Knowledge of using the `pool.ntp.org` service.

- Awareness of the `ntpq` command.

Terms and Utilities

- `/usr/share/zoneinfo/`
- `/etc/timezone`
- `/etc/localtime`
- `/etc/ntp.conf`
- `/etc/chrony.conf`
- `date`
- `hwclock`
- `ntpd`
- `ntpdate`
- `chronyc`
- `pool.ntp.org`

How a computer keeps its time

There is a clock in your computer; a hardware clock on your motherboard! It has its own battery and keeps the time even when the computer is off. When the system boots, the OS reads this **hardware time** and it sets it's own **system time** based on the hardware clock and uses this clock whenever it needs to know the time.

Hardware clock can be the set on localtime (the time shown on your clock wherever you are) or UTC time (standard time). The `hwclock` can be used to show the time based on the hwtime. See how it works based on the hardware time even after we DO CHANGE the system time:

Copy

```
$ date
Fri Jun 23 01:47:22 PM +0330 2023
$ sudo date -s "Jan 22 22:22:22 2022"
Sat Jan 22 10:22:22 PM +0330 2022
$ date
Sat Jan 22 10:22:29 PM +0330 2022
$ sudo hwclock
2023-06-23 13:47:41.160122+03:30
```

> Even when the hardware clock is set on UTC, `hwclock` date shows the date in the localtime (time after adding the timezone to the UTC time)

Older OSs used to set the hardware clock on localtime zone instead of timezone. This can be achieved by:

Copy

```
# hwclock --localtime --set --date="01/05/2023 22:04:00"
```

The previous commands sets the hardware clock on that specific date and tell it that this is the localtime. If you want to change back to UTC time, issue:

Copy

```
# hwclock -u -w
```

Here, `-w` tells the `hwclock` to set the hardware time based on the current system time and `-u` tells the hwclock that we are using the UTC. This also sets the HWClock using UTC int he `\etc\adjtime` file.

If you set a time on the hardware clock without mentioning it being UTC / Local, the `/etc/adjtime` will decide this, if this file does not exists, the UTC will be used.

You already know about the `timedatectl` and `date` from the Localization and globalization chapter so I wont repeat them here.

Time Zones

We have seen time zones in previous chapters. But in short, there are two important files here.

The `/etc/timezone` is a text file, representing your timezone. This is used if a program wants to know about your time zone of show it. For example:

Copy

```
$ cat /etc/timezone
Asia/Tehran
```

But the `/etc/localtime` is a binary file describing your timezone info to the system (for example the `date` file):

Copy

```
$ ls -l /etc/localtime
lrwxrwxrwx 1 root root 31 Jun 22 11:19 /etc/localtime -> /usr/share/zoneinfo/Asia/Tehran
$ file /usr/share/zoneinfo/Asia/Tehran
/usr/share/zoneinfo/Asia/Tehran: timezone data, version 2, no gmt time flags, no std time flags, no leap seconds, 72 transition times, 8 abbr
```

In many cases, this is a soft link to a file located at `/usr/share/zoneinfo/` so it will be updated in case of system update.

NTP

Network Time Protocol is my personal favorite protocol. It is one of the coolest protocols ever if you dive into its details. But unfortunately for LPIC1 you do not need to dive into NTP depths. This protocol uses NTP servers to find out the accurate time shown by best atomic clocks on this planet. One of the most famous severs used by ntp people is `pool.ntp.org`. If you check that website you will see that it is a **pool** of ntp servers and by giving your NTP server the `pool.ntp.org`, it will be redirected to one of the many ntp servers available on that pool.

ntpdate

The most straight forward command to set the systemclock is `ntpdate` and used like this:

Copy

```
$ sudo ntpdate pool.ntp.org
23 Jun 14:49:55 ntpdate[160138]: adjust time server 31.214.170.254 offset +0.020196 sec
```

After this, we need to set the hwclock to the recently synced system time using `sudo hwclock -w`.

ntpd

Instead of manually setting the time each time, you can use a linux service called `ntp` to keep your time using some time servers (the most famous one is pool.ntp.org). Install the `ntp` and start the server:

In Debian/Ubuntu:

Copy

```
# apt install ntp
# systemctl start ntp
```

Fun fact? you can not use both! Look at this:

Copy

```
root@funlife:~# ntpdate pool.ntp.org
23 Jun 14:49:55 ntpdate[18670]: the NTP socket is in use, exiting
```

As you can see, now the `ntp` is using the NTP port and `ntpdate` has problems starting up.

Main configuration file of `ntp` is located at `/etc/ntp.conf`:

Copy

```
# cat /etc/ntp.conf
# /etc/ntp.conf, configuration for ntpd; see ntp.conf(5) for help

driftfile /var/lib/ntp/ntp.drift

# Enable this if you want statistics to be logged.
#statsdir /var/log/ntpstats/

statistics loopstats peerstats clockstats
filegen loopstats file loopstats type day enable
filegen peerstats file peerstats type day enable
filegen clockstats file clockstats type day enable


# You do need to talk to an NTP server or two (or three).
#server ntp.your-provider.example

# pool.ntp.org maps to about 1000 low-stratum NTP servers.  Your server will
# pick a different set every time it starts up.  Please consider joining the
# pool: <http://www.pool.ntp.org/join.html>
pool 0.debian.pool.ntp.org iburst
pool 1.debian.pool.ntp.org iburst
pool 2.debian.pool.ntp.org iburst
pool 3.debian.pool.ntp.org iburst


# Access control configuration; see /usr/share/doc/ntp-doc/html/accopt.html for
# details.  The web page <http://support.ntp.org/bin/view/Support/AccessRestrictions>
# might also be helpful.
#
# Note that "restrict" applies to both servers and clients, so a configuration
# that might be intended to block requests from certain clients could also end
# up blocking replies from your own upstream servers.

# By default, exchange time with everybody, but don't allow configuration.
restrict -4 default kod notrap nomodify nopeer noquery limited
restrict -6 default kod notrap nomodify nopeer noquery limited

# Local users may interrogate the ntp server more closely.
restrict 127.0.0.1
restrict ::1

# Needed for adding pool entries
restrict source notrap nomodify noquery

# Clients from this (example!) subnet have unlimited access, but only if
# cryptographically authenticated.
#restrict 192.168.123.0 mask 255.255.255.0 notrust


# If you want to provide time to your local subnet, change the next line.
# (Again, the address is an example only.)
#broadcast 192.168.123.255

# If you want to listen to time broadcasts on your local subnet, de-comment the
# next lines.  Please do this only if you trust everybody on the network!
#disable auth
#broadcastclient
```

If needed, you can change the ntp servers to the ones you want to use.

Review the configuration and you will see cool things like providing your ntp service to other computers although you do not need these for passing LPIC.

ntpq

The `ntpq` queries the ntp service. One famous switch is `-p` (for Print) that shows the pool we are using to sync the clock:

Copy

```
 ntpq -p
     remote           refid      st t when poll reach   delay   offset  jitter
==============================================================================
 0.debian.pool.n .POOL.          16 p    -   64    0    0.000    0.000   0.000
 1.debian.pool.n .POOL.          16 p    -   64    0    0.000    0.000   0.000
 2.debian.pool.n .POOL.          16 p    -   64    0    0.000    0.000   0.000
 3.debian.pool.n .POOL.          16 p    -   64    0    0.000    0.000   0.000
+46.209.14.1     192.168.5.2      4 u    7   64    1   58.300  -15.546  14.519
-ntp.tums.ac.ir  195.161.115.4    4 u    4   64    1   30.636    2.485   4.025
*194.225.150.25  194.190.168.1    2 u    5   64    1   31.478   -3.870  95.635
+5.160.24.41     192.168.5.2      4 u    3   64    1   90.000  -28.328  21.643
```

In this output a `*` means that the ntp is using this server as the main reference, `+` means that this is a good server and `-` shows an out of range server which will be neglected.

chrony

Another and a newer NTP protocol implemenation is the `chrony`. Compared to `ntpd` , this tool provides better results at synchronize time difficult conditions such as intermittent network connections (such as laptops) and congested networks. Chrony is the default NTP client in RedHat 8, SUSE 15 and many other distributions. Another adavn

Here is a sample `chrony.conf` file:

Copy

```
$ cat /etc/chrony/chrony.conf
# Welcome to the chrony configuration file. See chrony.conf(5) for more
# information about usable directives.

# Include configuration files found in /etc/chrony/conf.d.
confdir /etc/chrony/conf.d

# This will use (up to):
# - 4 sources from ntp.ubuntu.com which some are ipv6 enabled
# - 2 sources from 2.ubuntu.pool.ntp.org which is ipv6 enabled as well
# - 1 source from [01].ubuntu.pool.ntp.org each (ipv4 only atm)
# This means by default, up to 6 dual-stack and up to 2 additional IPv4-only
# sources will be used.
# At the same time it retains some protection against one of the entries being
# down (compare to just using one of the lines). See (LP: #1754358) for the
# discussion.
#
# About using servers from the NTP Pool Project in general see (LP: #104525).
# Approved by Ubuntu Technical Board on 2011-02-08.
# See http://www.pool.ntp.org/join.html for more information.
pool ntp.ubuntu.com        iburst maxsources 4
pool 0.ubuntu.pool.ntp.org iburst maxsources 1
pool 1.ubuntu.pool.ntp.org iburst maxsources 1
pool 2.ubuntu.pool.ntp.org iburst maxsources 2

# Use time sources from DHCP.
sourcedir /run/chrony-dhcp

# Use NTP sources found in /etc/chrony/sources.d.
sourcedir /etc/chrony/sources.d

# This directive specify the location of the file containing ID/key pairs for
# NTP authentication.
keyfile /etc/chrony/chrony.keys

# This directive specify the file into which chronyd will store the rate
# information.
driftfile /var/lib/chrony/chrony.drift

# Save NTS keys and cookies.
ntsdumpdir /var/lib/chrony

# Uncomment the following line to turn logging on.
#log tracking measurements statistics

# Log files location.
logdir /var/log/chrony
```

```
# Stop bad estimates upsetting machine clock.
maxupdateskew 100.0

# This directive enables kernel synchronization (every 11 minutes) of the
# real-time clock. Note that it can’t be used along with the 'rtcfile' directive.
rtcsync

# Step the system clock instead of slewing it if the adjustment is larger than
# one second, but only in the first three clock updates.
makestep 1 3

# Get TAI-UTC offset and leap seconds from the system tz database.
# This directive must be commented out when using time sources serving
# leap-smeared time.
leapsectz right/UTC
```

To control the `chrony` service, there is a CLI (Command Line Interface) which is called `chronyc`. It it used to monitor chronyd's performance and to change various operating parameters. If a "command" is passed to the `chronyc`, the results will be shown, otherwise you will get a command to issue your commands. Have a look:

Copy

```
$ chronyc tracking
Reference ID    : 0FED61D6 (paris.time.system76.com)
Stratum         : 3
Ref time (UTC)  : Fri Jun 23 12:44:20 2023
System time     : 0.001574947 seconds slow of NTP time
Last offset     : +0.000513619 seconds
RMS offset      : 0.065126784 seconds
Frequency       : 8.705 ppm slow
Residual freq   : +0.278 ppm
Skew            : 14.972 ppm
Root delay      : 0.180896595 seconds
Root dispersion : 0.013603540 seconds
Update interval : 128.6 seconds
Leap status     : Normal
$ chronyc
chrony version 4.2
Copyright (C) 1997-2003, 2007, 2009-2021 Richard P. Curnow and others
chrony comes with ABSOLUTELY NO WARRANTY.  This is free software, and
you are welcome to redistribute it under certain conditions.  See the
GNU General Public License version 2 for details.

chronyc> activity
200 OK
12 sources online
0 sources offline
0 sources doing burst (return to online)
0 sources doing burst (return to offline)
3 sources with unknown address

chronyc> sources
MS Name/IP address         Stratum Poll Reach LastRx Last sample
===============================================================================
^? prod-ntp-4.ntp4.ps5.cano>     2   7   375    20  +5039us[+5039us] +/-   85ms
^? prod-ntp-5.ntp4.ps5.cano>     2   7   377    83    +12ms[  +15ms] +/-   87ms
^? prod-ntp-3.ntp4.ps5.cano>     2   7   177    22    +13ms[  +17ms] +/-   91ms
^? alphyn.canonical.com          2   7   277    22  +3298us[+6616us] +/-  185ms
^? mail.stumpflee.com            2   7   373    19  +6221us[+6221us] +/-  117ms
^? meetbsd.ir                    2   6   377    27    -23ms[  -20ms] +/-  231ms
^? 188.121.119.122               3   7   377    26    -27ms[  -23ms] +/-  131ms
^+ brazil.time.system76.com      2   7   377    20    -13ms[  -13ms] +/-  190ms
^+ ohio.time.system76.com        2   7   377    21    +23ms[  +23ms] +/-  171ms
^+ oregon.time.system76.com      2   7   377    23    +22ms[  +26ms] +/-  217ms
^* paris.time.system76.com       2   7   377    24  -4880us[-1560us] +/-  109ms
^+ virginia.time.system76.c>     2   7   163    19  +5465us[+5465us] +/-  148ms

chronyc> exit
$ sudo chronyc makestep
200 OK
```

`chrnoyc` connects to the chrony service using TCP or Unix sockets. Therefore, it is possible to use a local `chronyc` to connect to remote `chrony` and issue commands, although in this case you are limited to mainly monitoring commands for security reasons.

108.2 System logging

Weight: 4

Candidates should be able to configure rsyslog. This objective also includes configuring the logging daemon to send log output to a central log server or accept log output as a central log server. Use of the systemd journal subsystem is covered. Also, awareness of syslog and syslog-ng as alternative logging systems is included.

Key Knowledge Areas

- Basic configuration of rsyslog.
- Understanding of standard facilities, priorities and actions.
- Query the systemd journal.
- Filter systemd journal data by criteria such as date, service or priority.
- Configure persistent systemd journal storage and journal size.
- Delete old systemd journal data.
- Retrieve systemd journal data from a rescue system or file system copy.
- Understand interaction of rsyslog with systemd-journald.
- Configuration of logrotate.
- Awareness of syslog and syslog-ng.

Terms and Utilities

- `/etc/rsyslog.conf`
- `/var/log/`
- `logger`
- `logrotate`
- `/etc/logrotate.conf`
- `/etc/logrotate.d/`
- `journalctl`
- `systemd-cat`
- `/etc/systemd/journald.conf`
- `/var/log/journal/`

History

Logs are an important part of Unix philosophy and design. The Kernel, services, programs and most events (let it be a crash or a login attempt) will generate logs. These logs can be examines / monitored to gain insights about the system and its status. If you watch a Linux guy on a windows machine for 1 hour, you will hear "where are the logs?".

If every single program tries to handle its own logs we will face 2 issues (at least!); 1. programming will be difficult because you need to invent (or import) the wheel and 2. different programmers will generate different kinds, formats and location for their logs and it will be difficult to keep them under your control. To overcome this, Unix world had its own solution: central logging service.

The older utility to manage logs was called `syslog`, then we had `syslog-ng` (new-generation) and later `rsyslog` which is what most distros were using till a couple of years ago. `rsyslog` is still present and is part of the GNU/Linux universe. It can receive logs from different programs (even over network) and save them in different places (files or files in network or even run some actions on them). It's logs are saved in the text format and can be managed by the tool of your choice, let it be `grep`, `less`, `tail`, `zless` (less a compressed file without explicitly opening it), `zcat` (catting a file without explicitly opening it), `grep`, ...

But these days `systemd` is conquering most of the linux world and logs are not an exception. SystemDs logging service is called `journald` and we use the `journalctl` command to read its logs; unfortunately they are not text files anymore.

You should also know about a couple of other tools (like `logrotate` to cleanup the older logs) and be familiar with some of more important system logs under GNU/Linux. So lets start.

> Its fun to know how the logs work in more depth. Here it is: whoever wants to log something eventually sends its messages to the `/dev/log` or `/dev/kmsg` devices (often by using a helper tool). The logging tool (say `rsyslog`) will read from these devices and process the logs based on its configs; and now, the messages are logs!

dmesg

As I told you, Linux loves logs and will be happy logging everything. But what about the boot process? What happens if the kernel wants to log something before the disks are ready? These logs are saved by the Kernel in the *Kernel Ring Buffer*. You can access it using the `dmesg` command.

log rotation

So logs are being generated and saved! Someone need to clean them up to prevent the disk from getting full. The `logrotate` utility takes care of this task. The main configuration is at `/etc/logrotate.conf`:

Copy

```
root@debian:# cat /etc/logrotate.conf
# see "man logrotate" for details

# global options do not affect preceding include directives

# rotate log files weekly
weekly

# keep 4 weeks worth of backlogs
rotate 4

# create new (empty) log files after rotating old ones
create

# use date as a suffix of the rotated file
#dateext

# uncomment this if you want your log files compressed
#compress

# packages drop log rotation information into this directory
include /etc/logrotate.d

# system-specific logs may also be configured here.
```

and specific services do create their logs in `/etc/logrotate.d/`:

Copy

```
root@debian:# cat /etc/logrotate.d/apt
/var/log/apt/term.log {
  rotate 12
  monthly
  compress
  missingok
  notifempty
}

/var/log/apt/history.log {
  rotate 12
  monthly
  compress
  missingok
  notifempty
}
```

These are the meaning of some of these parameters:

parameter	meaning
weekly	rotate logs weekly
missingok	it is fine if there is no log for this week
rotate 52	keep the latest 52 logs and delete the older ones
compress	compress the logs

parameter	meaning
create 0640 www-data adm	create the files with this access and owners
pre & post rotate	run these scripts or commands before and after the rotation

This above configuration will create a zipped file for each week, keeping only 52 of them instead of a huge log file for this program.

The `logrotate` runs using crons and does its job on a daily basis based on a configuration on `/etc/cron.daily/logrotate`

some famous log files

Generally logs are saved at `/var/log`. In case of any issue and if you do not know what to do, it is a common practice to run a `ls -ltrh /var/log/` to see if any program generated a new log or not.

But lets have a look at some of the famous logs:

`/var/log/auth.log` (in Debian Based)

Authentication processes will log here. Things like `cron` jobs, failed logins, `sudo` informations, ...

`/var/log/syslog`

A centralized place most of the logs received by `rsyslogd` if a specific logfile is not provided in `/etc/rsyslog.conf`

`/var/log/debug`

Debug information from programs.

`/var/log/kern.log`

Kernel messages.

`/var/log/messages`

Informative messages from services. This is also the default place for logs for remote clients.

`/var/run/utmp` & `/var/log/wtmp`

Successful logins.

`/var/log/btmp`

Failed logins. You can check this to see if anyone is trying to guess your passwords!

`/var/log/faillog`

Failed authentication attempts.

`/var/log/lastlog`

Date and time of recent user logins.

Service logs

Service logs do create files or directories at `/var/log` and update their logs there. For example there might be a `/var/log/apache2/` (or `/var/log/httpd`) for the Apache HTTP server or a `/var/log/mysql` for the MySQL DB.

rsyslog

This is the newer generation of syslogs and is used in many environments. Its main configuration is located at `/etc/rsyslog.conf` and it also reads anything in `/etc/rssylog.d/` directory. Its configuration consists of 3 main sections:

1. `MODULES`, modules used. For example to let the rsyslog use UDP connections
2. `GLOBAL DIRECTIVES`, general configurations like directories accesses
3. `RULES`, A combination of *facilities*, *priorities* and *actions* tells the `rsyslog` what to do with each log.

Copy

```
root@debian:~# cat /etc/rsyslog.conf
# /etc/rsyslog.conf configuration file for rsyslog
#
# For more information install rsyslog-doc and see
# /usr/share/doc/rsyslog-doc/html/configuration/index.html


#################
#### MODULES ####
#################

module(load="imuxsock") # provides support for local system logging
module(load="imklog")   # provides kernel logging support
#module(load="immark")  # provides --MARK-- message capability

# provides UDP syslog reception
#module(load="imudp")
#input(type="imudp" port="514")

# provides TCP syslog reception
#module(load="imtcp")
#input(type="imtcp" port="514")


###########################
#### GLOBAL DIRECTIVES ####
###########################

#
# Set the default permissions for all log files.
#
$FileOwner root
$FileGroup adm
$FileCreateMode 0640
$DirCreateMode 0755
$Umask 0022

#
# Where to place spool and state files
#
$WorkDirectory /var/spool/rsyslog

#
# Include all config files in /etc/rsyslog.d/
#
$IncludeConfig /etc/rsyslog.d/*.conf


###############
#### RULES ####
```

```
###############

#
# Log anything besides private authentication messages to a single log file
#
*.*;auth,authpriv.none      -/var/log/syslog

#
# Log commonly used facilities to their own log file
#
auth,authpriv.*         /var/log/auth.log
cron.*              -/var/log/cron.log
kern.*              -/var/log/kern.log
mail.*              -/var/log/mail.log
user.*              -/var/log/user.log

#
# Emergencies are sent to everybody logged in.
#
*.emerg             :omusrmsg:*
```

As you can see **facilities** could be things like:

> `kern`, `user`, `mail`, `daemon`, `cron`, `auth`, `ntp`, `security`, `console`, `syslog`, **...**

The **priority** could be one of the below:

> `emerg`/`panic`, `alert`, `crit`, `err`/`error`, `warn`/`warning`, `notice`, `info` or `debug`.

On the **action** part we can have things like these:

action	sample	meaning
filename	/usr/log/logins.log	will write the log to this file
username	jadi	will notify that person on the screen
@ip	@192.168.1.100	will send this log to this log server and that log server will decide what to do with it based on its configs

So a line like this will show the kernel panics to a remote log server and also will log everything on every level to a log file:

Copy

```
kern.panic      @192.168.1.100
*.*             /var/log/messages
```

If you log some specific priority, all the **more important** things will be logged too! So if you write `cron.notice /var/log/cron/cron.log`, you are logging the `emerg/panic`, `alert`, `critical`, `error`, `warning` and `notice` logs of the cron category too.

Priority Levels (According to Syslog(3) and logger manpage):

Copy

```
       This determines the importance of the message.  The levels are, in order of decreasing importance:

       emerge      system is unusable

       alert      action must be taken immediately

       crit       critical conditions

       error        error conditions

       warning    warning conditions

       notice     normal, but significant, condition

       info       informational message

       debug      debug-level message


       panic      deprecated synonym for emerg
       error      deprecated synonym for err
       warn       deprecated synonym for warning
```

> If you need to log ONLY one specific level, add an equal sign (=) before the priority like this `local3.=alert /var/log/user.alert.log`.

It is important to know that the binary which logs the **kern* category is a standalone daemon. This daemon is called `klogd` and uses same configuration files. Why? so even after everything is crashed, `klogd` will be able to log the kernel's crashes ;).

logger

If you need to send something toward the `rsyslog`, you can use the `logger` tool.

Copy

```
root@debian:# logger Testing my lovely tool
root@debian:# logger local1.emerg Nothing emergent for sure
root@debian:# tail -3 /var/log/syslog
2023-06-30T06:05:01.325358-04:00 debian CRON[914]: (root) CMD (command -v debian-sa1 > /dev/null && debian-sa1 1 1)
2023-06-30T06:13:27.671410-04:00 debian root: Testing my lovely tool
2023-06-30T06:13:45.795158-04:00 debian root: local1.emerg Nothing emergent for sure
```

journald

You have seen the `systemd` in various parts of this course. We have seen how it is part of the init process and how it handles the services and timers. All of these activities (and other logs reaching system journalying system) are logged in binary files, readable using `journalctl` which is part of the `systemd-journald` utility.

Copy

```
root@debian:~# systemctl status systemd-journald
● systemd-journald.service - Journal Service
     Loaded: loaded (/lib/systemd/system/systemd-journald.service; static)
     Active: active (running) since Fri 2023-06-30 05:28:50 EDT; 52min ago
TriggeredBy: ● systemd-journald-dev-log.socket
             ● systemd-journald-audit.socket
             ● systemd-journald.socket
       Docs: man:systemd-journald.service(8)
             man:journald.conf(5)
   Main PID: 261 (systemd-journal)
     Status: "Processing requests..."
      Tasks: 1 (limit: 4583)
     Memory: 17.1M
        CPU: 138ms
     CGroup: /system.slice/systemd-journald.service
             └─261 /lib/systemd/systemd-journald

Jun 30 05:28:50 debian systemd-journald[261]: Journal started
Jun 30 05:28:50 debian systemd-journald[261]: Runtime Journal (/run/log/journal/ec22e43962c64359b9b25cfa650b025b) is 4.9M, max 39.1M,>
Jun 30 05:28:50 debian systemd-journald[261]: Time spent on flushing to /var/log/journal/ec22e43962c64359b9b25cfa650b025b is 23.816ms>
Jun 30 05:28:50 debian systemd-journald[261]: System Journal (/var/log/journal/ec22e43962c64359b9b25cfa650b025b) is 28.2M, max 4.0G, >
Jun 30 05:28:50 debian systemd-journald[261]: Received client request to flush runtime journal.
Notice: journal has been rotated since unit was started, output may be incomplete.
```

and here is the configuration file:

Copy

```
root@debian:~# cat /etc/systemd/journald.conf
#  This file is part of systemd.
#
#  systemd is free software; you can redistribute it and/or modify it under the
#  terms of the GNU Lesser General Public License as published by the Free
#  Software Foundation; either version 2.1 of the License, or (at your option)
#  any later version.
#
# Entries in this file show the compile time defaults. Local configuration
```

```
# should be created by either modifying this file, or by creating "drop-ins" in
# the journald.conf.d/ subdirectory. The latter is generally recommended.
# Defaults can be restored by simply deleting this file and all drop-ins.
#
# Use 'systemd-analyze cat-config systemd/journald.conf' to display the full config.
#
# See journald.conf(5) for details.

[Journal]
#Storage=auto
#Compress=yes
#Seal=yes
#SplitMode=uid
#SyncIntervalSec=5m
#RateLimitIntervalSec=30s
#RateLimitBurst=10000
#SystemMaxUse=
#SystemKeepFree=
#SystemMaxFileSize=
#SystemMaxFiles=100
#RuntimeMaxUse=
#RuntimeKeepFree=
#RuntimeMaxFileSize=
#RuntimeMaxFiles=100
#MaxRetentionSec=
#MaxFileSec=1month
#ForwardToSyslog=yes
#ForwardToKMsg=no
#ForwardToConsole=no
#ForwardToWall=yes
#TTYPath=/dev/console
#MaxLevelStore=debug
#MaxLevelSyslog=debug
#MaxLevelKMsg=notice
#MaxLevelConsole=info
#MaxLevelWall=emerg
#LineMax=48K
#ReadKMsg=yes
#Audit=no
```

querying journal contents

If you run the `journalctl` you will get all the logs... too much. So there are some useful switches:

Switch	Meaning
-r	show in reverse; newer on top
-f	keep showing the tail
-e	show and go to the end
-n	show this many lines from the end (newest ones)
-k	show kernel message (equal to `dmesg`)
-b	show from a specific boot, -1 is the previous one, 0 is the current one. You can check the list using `--list-boots`
-p	from specific priority, say `-p err`
-u	from a specific systemd unit

You can also use the `--since` and `--until` to show a specific time range, It is possible to provide time as `YYYY-MM-DD HH:MM:SS` or use things like `yesterday`, `today`, `tomorrow`, `now` or even `--since "10 minutes ago"` which is equals to `--since "-2 minutes"`.

If there are too many lines, push `>` to jump to the end (or `<` to go to the beginning)

You can also add the name of the program to the command to see the logs from that specific program; say `journalctl /usr/bin/xrdp` or use some fields like `PRIORITY=`, `_PID=` and provide values.

systemd-cat

This tools is used when you want to manually send logs to the journaling system. It sends its input to the journal or if provided, runs the command and sends the result to the journal.

Copy

```
root@debian:~# echo "This is my first test" | systemd-cat
root@debian:~# systemd-cat -p info uptime #sending priority too
root@debian:~# journalctl -n 3
Jun 30 06:31:01 debian systemd[1]: Finished apt-daily.service - Daily apt download activities.
Jun 30 06:34:18 debian cat[1020]: This is my first test
Jun 30 06:34:48 debian uptime[1022]:  06:34:48 up  1:05,  4 users,  load average: 0.00, 0.00, 0.00
```

managing storage

The systemd journals can keep their logs in memeory or write them to the disk or drop all the logs. This is determined by the configurations in `/etc/systemd/journald.conf`. But the default behaviour is as follow:

1. The system checks the `/var/log/journal`. If this directory, exists logs will be saved in a directory inside this. The name of the directory will be decided by looking at `/etc/machine-id`
2. If the `/var/log/journal` is not there, the logs will be saved in memory at `/run/log/jouranl` and in the directory decided by `/etc/machine-id`.

But you can overcome these configs by the following sections:

Copy

```
Storage=volatile     # keep the logs in memory
Storage=persistent   # keep on disk, if needed create the directories
Storage=auto         # will write to disk only if the directory exists
Storage=none         # do not keep logs
```

If the logs are being written to disk, these variable will manage the disk usage:

Variable Name	Usage
SystemMaxUse	The Maximum disk usage, say 500M. The default is on `10%`
SystemKeepFree	Keep at least this much free, say 1G. The default is 15%
SystemMaxFileSize	Maximum size of each individual file. The default is 1/8 of SystemMaxUse
SystemMaxFiles	Max number of non-currently-active files. The default is 100

If you are keeping the logs in **memory**, the equivalent variables are `RuntimeMaxUse`, `RuntimeKeepFree`, `RuntimeMaxFileSize`, & `RuntimeMaxFiles`.

In case you need to do the clean-up manually (with the help of systemd tools for sure), you can run `journalctl` with these switches:

Switch	Usage
--vacuum-time	clean everything older than this. for example `--vacuum-time=3months` clean anything older than 3 months. You can use `s`, `m`, `h` for seconds, minutes and days and `d`/`days`, `months`, `weeks`/`w` and `years`/`y`.
--vacuum-size	eliminate till logs occupy a specific size; say 1G
--vacuum-files	Only keep this much archive files

checking logs from a recovered system

If you have a crashed / non-booting system, you can still examine its logs; if the files are there. As you know the files are in `/var/log/journal/{mchine-id}` where machine-id is in `/etc/machine-id` of the crashed machine.

You can move these files to a directory after booting the crashed machine with a live linux or mount its hard on another machine or even examine them in place. The `-D` (or `--directory`) switch of `journalctl` indicates the location of the journal files. So if your crashed machine id is `ec22e43962c64359b9b25cfa650b025b` and you've mounted its `/var/` under your `/mnt/var/` directory, you can issue this command to read its logs and see what had happened:

Copy

```
journalctl -D=/mnt/var/log/journal/ec22e43962c64359b9b25cfa650b025b/
```

You can also use the `--merge` switch to merge these logs into your machine or use `--file` to check only one specific journal file. Lastly if the exact location of journal files are not known, you can use `--root /mnt` and tell the `journalctl` to search there for journal files.

108.3 Mail Transfer Agent (MTA) basics

Weight: 3

Candidates should be aware of the commonly available MTA programs and be able to perform basic forward and alias configuration on a client host. Other configuration files are not covered.

Key Knowledge Areas

- Create e-mail aliases.

- Configure e-mail forwarding.
- Knowledge of commonly available MTA programs (`postfix`, `sendmail`, `exim`) (no configuration)

Terms and Utilities

- `~/.forward`
- sendmail emulation layer commands
- `newaliases`
- `mail`
- `mailq`
- `postfix`
- `sendmail`
- `exim`

MTAs

Email is an integral part of many GNU/Linux and Unix systems. Each user do have a mail box and can send / receive email to other local users. This is done via MTAs (Mai Transfer Agents). In other words, MTAs are programs which handle emails in your operating system. They can receive and dispatch emails locally and over the network. There are different options for MTAs. In this section we will do a quick review on them and you will see how you can send emails to other uses (or over the internet) and how you can check your local mails.

sendmail

Is one of the oldest options available. It is huge and kind of difficult to configure and not that security oriented. Because of these, few systems use it as default their MTA.

exim

It aims to be a general and flexible mailer with extensive facilities for checking incoming e-mail. It is feature rich with ACLs, authentication and many other features.

postfix

This is a new alternative to `sendmail` and uses easy to understand configuration files. It supports multiple domains, encryption, etc. Postfix is what you see on most distros as the default MTA.

> Most desktop distros do not install MTAs by default. If you want, I suggest installing the `postfix` (and `mailx` or `bsd-mailx`) via your package manager.

sendmail emulation layer

As you already know, `sendmail` is the oldest MTA alive and therefore, many other MTAs try to comply with it and has a *sendmail emulation layer* to keep themselves backward compatible with sendmail. Thats why you can type `sendmail` on whatever distro you are or use the `mailq` and check your mail regardless of your MTA choice.

aliases

There are some mail aliases on the system. Defined in `/etc/aliases`.

Copy

```
$ cat /etc/aliases
#
#  Aliases in this file will NOT be expanded in the header from
#  Mail, but WILL be visible over networks or from /bin/mail.
#
#       >>>>>>>>>>      The program "newaliases" must be run after
#       >> NOTE >>      this file is updated for any changes to
#       >>>>>>>>>>      show through to sendmail.
#

# Basic system aliases -- these MUST be present.
mailer-daemon:  postmaster
postmaster:     root

# General redirections for pseudo accounts.
bin:            root
daemon:         root
adm:            root
lp:             root
sync:           root
shutdown:       root
halt:           root
mail:           root
news:           root
uucp:           root
operator:       root
games:          root
www:            webmaster
webmaster:      root
[ .... ]
```

This tells the system if there is a message for *news*, it should be delivered to *root* and if the email is written to *www* it should be delivered to *webmaster*.

In case of any change in this file, you need to run the `newaliases` command.

sending mail

It is possible to send an email from the command line using the `mail` command:

Copy

```
[jadi@funlife ~]$ mail news
Subject: Email to news user
hahah.. we know where this will go.
this will go to root and then to jadi!

Hi Jadi!

Cc:
[jadi@funlife ~]$ mail
Mail version 8.1.2 01/15/2001.  Type ? for help.
"/var/mail/jadi": 12 messages 12 new
>N  1 root@funlife       Sat Jan 02 08:50   39/1373  apt-listchanges: news for f
 N  2 root@funlife       Sat Jan 02 09:01  165/7438  apt-listchanges: news for f
 N  3 jadi@funlife       Sat Jan 02 19:58   18/640   *** SECURITY information fo
 N  4 jadi@funlife       Sat Jan 02 20:04   18/631   *** SECURITY information fo
 N  5 jadi@funlife       Sun Jan 03 10:15   18/664   *** SECURITY information fo
 N  6 root@funlife       Mon Jan 04 12:42   27/941   Cron <jadi@funlife> /home/j
 N  7 root@funlife       Mon Jan 04 17:11   26/845   apt-listchanges: news for f
 N  8 root@funlife       Tue Jan 05 18:42   27/945   Cron <jadi@funlife> /home/j
 N  9 root@funlife       Wed Jan 06 09:17   46/1788  apt-listchanges: news for f
 N 10 root@funlife       Thu Jan 07 12:42   27/945   Cron <jadi@funlife> /home/j
 N 11 root@funlife       Thu Jan 07 18:42   27/943   Cron <jadi@funlife> /home/j
 N 12 jadi@funlife       Thu Jan  7 19:53   17/478   Email to news user
& 12
Message 12:
From jadi@funlife  Thu Jan  7 19:53:08 2016
X-Original-To: news
To: news@funlife
Subject: Email to news user
Date: Thu,  7 Jan 2016 19:53:08 +0330 (IRST)
From: jadi@funlife (jadi)

hahah.. we know where this will go.
this will go to root and then to jadi!

Hi Jadi!

& d
```

```
& q
Held 11 messages in /var/mail/jadi
```

local forwards

We saw that it is possible to forward emails using the `/etc/aliases`. That file is not writable by normal users so what a normal user like *jadi* should do?

Each user can create a `.forward` file in her own directory and all mail targeted to that user will be forwarded to that address.

> You can even put a complete email address like `jadijadi@gmail.com` in your `.forward` file.

It also can send email from the command line or even within your scripts by issuing something like `echo -e "email content" | mail -s "email subject" "example@example.com"`.

mailq

This command lists the mail queue. Each entry shows the queue file ID, message size, arrival time, sender, and the recipients that still need to be delivered. If mail could not be delivered upon the last attempt, the reason for failure is shown. The sysadmin can use this command to check the status of emails still in the queues.

Copy

```
$ mailq
-Queue ID- --Size-- ----Arrival Time---- -Sender/Recipient-------
AA52C228E6B      468 Thu Jan  7 19:59:41  jadi@funlife
(connect to alt2.gmail-smtp-in.l.google.com[2404:6800:4003:c01::1a]:25: Network is unreachable)
                                         jadijadi@gmail.com

-- 0 Kbytes in 1 Request.
```

108.4 Manage printers and printing

Weight: 2

Candidates should be able to manage print queues and user print jobs using CUPS and the LPD compatibility interface.

Key Knowledge Areas

- Basic CUPS configuration (for local and remote printers).
- Manage user print queues.
- Troubleshoot general printing problems.
- Add and remove jobs from configured printer queues.

Terms and Utilities

- CUPS configuration files, tools and utilities
- `/etc/cups/`
- lpd legacy interface (lpr, lprm, lpq)

CUPS

Most Linux distributions use the CUPS package for printing. You may need to install it via your package manager and start its service using the systemd or the init system your system use.

Copy

```
$ sudo apt install cups
[sudo] password for jadi:
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following additional packages will be installed:
  acl avahi-daemon colord colord-data cups-browsed cups-client cups-common cups-core-drivers
  cups-daemon cups-filters cups-filters-core-drivers cups-ipp-utils cups-ppdc
  cups-server-common fonts-droid-fallback fonts-noto-mono fonts-urw-base35 ghostscript
  ipp-usb libavahi-core7 libavahi-glib1 libcolorhug2 libcupsfilters1 libdaemon0 libexif12
  libfontembed1 libgphoto2-6 libgphoto2-l10n libgphoto2-port12 libgs-common libgs10
  libgs10-common libgusb2 libidn12 libieee1284-3 libijs-0.35 libjbig2dec0 libjson-glib-1.0-0
  libjson-glib-1.0-common liblouis-data liblouis20 liblouisutdml-bin liblouisutdml-data
  liblouisutdml9 libltdl7 libnss-mdns libpaper-utils libpaper1 libpoppler-cpp0v5
  libpoppler-glib8 libpoppler126 libqpdf29 libsane-common libsane1 libsnmp-base libsnmp40
  lynx lynx-common mailcap poppler-data poppler-utils sane-airscan sane-utils update-inetd
  usb.ids
Suggested packages:
  avahi-autoipd colord-sensor-argyll cups-bsd cups-pdf foomatic-db-compressed-ppds
  | foomatic-db smbclient antiword docx2txt imagemagick fonts-noto fonts-freefont-otf
  | fonts-freefont-ttf fonts-texgyre gphoto2 ooo2dbk rtf2xml avahi-autoipd | zeroconf hplip
  snmp-mibs-downloader fonts-japanese-mincho | fonts-ipafont-mincho fonts-arphic-ukai
  fonts-arphic-uming fonts-nanum unpaper
The following NEW packages will be installed:
  acl avahi-daemon colord colord-data cups cups-browsed cups-client cups-common
  cups-core-drivers cups-daemon cups-filters cups-filters-core-drivers cups-ipp-utils
  cups-ppdc cups-server-common fonts-droid-fallback fonts-noto-mono fonts-urw-base35
  ghostscript ipp-usb libavahi-core7 libavahi-glib1 libcolorhug2 libcupsfilters1 libdaemon0
  libexif12 libfontembed1 libgphoto2-6 libgphoto2-l10n libgphoto2-port12 libgs-common
  libgs10 libgs10-common libgusb2 libidn12 libieee1284-3 libijs-0.35 libjbig2dec0
  libjson-glib-1.0-0 libjson-glib-1.0-common liblouis-data liblouis20 liblouisutdml-bin
  liblouisutdml-data liblouisutdml9 libltdl7 libnss-mdns libpaper-utils libpaper1
  libpoppler-cpp0v5 libpoppler-glib8 libpoppler126 libqpdf29 libsane-common libsane1
  libsnmp-base libsnmp40 lynx lynx-common mailcap poppler-data poppler-utils sane-airscan
  sane-utils update-inetd usb.ids
0 upgraded, 66 newly installed, 0 to remove and 7 not upgraded.
Need to get 40.7 MB of archives.
After this operation, 163 MB of additional disk space will be used.
Do you want to continue? [Y/n]
...
...
```

CUPS stands for Common Unix Printing System and as you can see, it installs many related packages and even suggests some more. This is because CUPS need lots of information about different printers and uses many tools to print. When installed, you need to start the service:

Copy

```
$ sudo systemctl start cups.service
$ sudo systemctl status cups.service
● cups.service - CUPS Scheduler
     Loaded: loaded (/lib/systemd/system/cups.service; enabled; preset: enabled)
     Active: active (running) since Sun 2023-07-16 13:50:20 EDT; 27s ago
TriggeredBy: ● cups.path
             ● cups.socket
       Docs: man:cupsd(8)
   Main PID: 2366 (cupsd)
     Status: "Scheduler is running..."
      Tasks: 1 (limit: 4583)
     Memory: 2.8M
        CPU: 400ms
     CGroup: /system.slice/cups.service
             └─2366 /usr/sbin/cupsd -l

Jul 16 13:50:20 debian systemd[1]: Starting cups.service - CUPS Scheduler...
Jul 16 13:50:20 debian systemd[1]: Started cups.service - CUPS Scheduler.
```

To access the CUPS services, there are different ways, including a web based interface, GUI programs on the graphical modes and even command line tools. CUPS is designed to be simple and able to use different printers from various vendors.

configuration files

As most other linux program, CUPS saves its configuration at `/etc` directory.

Copy

```
# ls /etc/cups/
cups-browsed.conf  cups-files.conf  ppd        raw.convs  snmp.conf  subscriptions.conf
cupsd.conf     interfaces       printers.conf  raw.types  ssl    subscriptions.conf.O
```

The main configuration file is the `cupsd.conf`. Take a look at it; it is very easy to understand. For example the `Listen localhost:631` line tells the CUPS to listen on localhost port 631. Here is a sample:

Copy

```
$ cat /etc/cups/cupsd.conf
#
# Configuration file for the CUPS scheduler.  See "man cupsd.conf" for a
# complete description of this file.
#

# Log general information in error_log - change "warn" to "debug"
# for troubleshooting...
LogLevel warn
PageLogFormat

# Specifies the maximum size of the log files before they are rotated.  The value "0" disables log rotation.
MaxLogSize 0

# Default error policy for printers
ErrorPolicy retry-job

# Only listen for connections from the local machine.
Listen localhost:631
Listen /run/cups/cups.sock

# Show shared printers on the local network.
Browsing Yes
BrowseLocalProtocols dnssd

# Default authentication type, when authentication is required...
DefaultAuthType Basic

# Web interface setting...
WebInterface Yes

# Timeout after cupsd exits if idle (applied only if cupsd runs on-demand - with -l)
IdleExitTimeout 60

# Restrict access to the server...
<Location />
  Order allow,deny
</Location>

# Restrict access to the admin pages...
<Location /admin>
  Order allow,deny
</Location>

# Restrict access to configuration files...
<Location /admin/conf>
  AuthType Default
  Require user @SYSTEM
  Order allow,deny
</Location>

# Restrict access to log files...
<Location /admin/log>
  AuthType Default
  Require user @SYSTEM
  Order allow,deny
</Location>

# Set the default printer/job policies...
<Policy default>
  # Job/subscription privacy...
  JobPrivateAccess default
  JobPrivateValues default
  SubscriptionPrivateAccess default
  SubscriptionPrivateValues default

  # Job-related operations must be done by the owner or an administrator...
  <Limit Create-Job Print-Job Print-URI Validate-Job>
    Order deny,allow
  </Limit>
```

```
    <Limit Send-Document Send-URI Hold-Job Release-Job Restart-Job Purge-Jobs Set-Job-Attributes Create-Job-Subscription Renew-Subscription Can
      Require user @OWNER @SYSTEM
      Order deny,allow
    </Limit>

    # All administration operations require an administrator to authenticate...
    <Limit CUPS-Add-Modify-Printer CUPS-Delete-Printer CUPS-Add-Modify-Class CUPS-Delete-Class CUPS-Set-Default CUPS-Get-Devices>
      AuthType Default
      Require user @SYSTEM
      Order deny,allow
    </Limit>

    # All printer operations require a printer operator to authenticate...
    <Limit Pause-Printer Resume-Printer Enable-Printer Disable-Printer Pause-Printer-After-Current-Job Hold-New-Jobs Release-Held-New-Jobs Deac
      AuthType Default
      Require user @SYSTEM
      Order deny,allow
    </Limit>

    # Only the owner or an administrator can cancel or authenticate a job...
    <Limit Cancel-Job CUPS-Authenticate-Job>
      Require user @OWNER @SYSTEM
      Order deny,allow
    </Limit>

    <Limit All>
      Order deny,allow
    </Limit>
  </Policy>

  # Set the authenticated printer/job policies...
  <Policy authenticated>
    # Job/subscription privacy...
    JobPrivateAccess default
    JobPrivateValues default
    SubscriptionPrivateAccess default
    SubscriptionPrivateValues default

    # Job-related operations must be done by the owner or an administrator...
    <Limit Create-Job Print-Job Print-URI Validate-Job>
      AuthType Default
      Order deny,allow
    </Limit>

    <Limit Send-Document Send-URI Hold-Job Release-Job Restart-Job Purge-Jobs Set-Job-Attributes Create-Job-Subscription Renew-Subscription Can
      AuthType Default
      Require user @OWNER @SYSTEM
      Order deny,allow
    </Limit>

    # All administration operations require an administrator to authenticate...
    <Limit CUPS-Add-Modify-Printer CUPS-Delete-Printer CUPS-Add-Modify-Class CUPS-Delete-Class CUPS-Set-Default>
      AuthType Default
      Require user @SYSTEM
      Order deny,allow
    </Limit>

    # All printer operations require a printer operator to authenticate...
    <Limit Pause-Printer Resume-Printer Enable-Printer Disable-Printer Pause-Printer-After-Current-Job Hold-New-Jobs Release-Held-New-Jobs Deac
      AuthType Default
      Require user @SYSTEM
      Order deny,allow
    </Limit>

    # Only the owner or an administrator can cancel or authenticate a job...
    <Limit Cancel-Job CUPS-Authenticate-Job>
      AuthType Default
      Require user @OWNER @SYSTEM
      Order deny,allow
    </Limit>

    <Limit All>
      Order deny,allow
    </Limit>
  </Policy>

  # Set the kerberized printer/job policies...
  <Policy kerberos>
    # Job/subscription privacy...
    JobPrivateAccess default
```

```
    JobPrivateValues default
    SubscriptionPrivateAccess default
    SubscriptionPrivateValues default

    # Job-related operations must be done by the owner or an administrator...
    <Limit Create-Job Print-Job Print-URI Validate-Job>
      AuthType Negotiate
      Order deny,allow
    </Limit>

    <Limit Send-Document Send-URI Hold-Job Release-Job Restart-Job Purge-Jobs Set-Job-Attributes Create-Job-Subscription Renew-Subscription Can
      AuthType Negotiate
      Require user @OWNER @SYSTEM
      Order deny,allow
    </Limit>

    # All administration operations require an administrator to authenticate...
    <Limit CUPS-Add-Modify-Printer CUPS-Delete-Printer CUPS-Add-Modify-Class CUPS-Delete-Class CUPS-Set-Default>
      AuthType Default
      Require user @SYSTEM
      Order deny,allow
    </Limit>

    # All printer operations require a printer operator to authenticate...
    <Limit Pause-Printer Resume-Printer Enable-Printer Disable-Printer Pause-Printer-After-Current-Job Hold-New-Jobs Release-Held-New-Jobs Deac
      AuthType Default
      Require user @SYSTEM
      Order deny,allow
    </Limit>

    # Only the owner or an administrator can cancel or authenticate a job...
    <Limit Cancel-Job CUPS-Authenticate-Job>
      AuthType Negotiate
      Require user @OWNER @SYSTEM
      Order deny,allow
    </Limit>

    <Limit All>
      Order deny,allow
    </Limit>
  </Policy>
  jadi@debian:~$
  Broadcast message from root@debian on pts/3 (Sun 2023-07-16 20:54:04 EDT):

  The system will power off now!

  Connection to 192.168.64.7 closed by remote host.
  Connection to 192.168.64.7 closed.
  ➜  lpic1book ssh jadi@192.168.64.7
  jadi@192.168.64.7's password:
  Linux debian 6.1.0-9-arm64 #1 SMP Debian 6.1.27-1 (2023-05-08) aarch64

  The programs included with the Debian GNU/Linux system are free software;
  the exact distribution terms for each program are described in the
  individual files in /usr/share/doc/*/copyright.

  Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
  permitted by applicable law.
  You have no mail.
  Last login: Mon Jul 17 08:13:13 2023
  jadi@debian:~$ sudo su -
  [sudo] password for jadi:
  root@debian:~# cd /etc/cups/
  root@debian:/etc/cups# ls
  cups-browsed.conf  cups-files.conf  ppd        raw.convs  snmp.conf  subscriptions.conf
  cupsd.conf     interfaces       printers.conf  raw.types  ssl    subscriptions.conf.O
  root@debian:/etc/cups# cd
  root@debian:~# ls /etc/cups/
  cups-browsed.conf  cups-files.conf  ppd        raw.convs  snmp.conf  subscriptions.conf
  cupsd.conf     interfaces       printers.conf  raw.types  ssl    subscriptions.conf.O
  root@debian:~# cat /etc/cups/cupsd.conf
  #
  # Configuration file for the CUPS scheduler.  See "man cupsd.conf" for a
  # complete description of this file.
  #

  # Log general information in error_log - change "warn" to "debug"
  # for troubleshooting...
  LogLevel warn
  PageLogFormat
```

```
# Specifies the maximum size of the log files before they are rotated.  The value "0" disables log rotation.
MaxLogSize 0

# Default error policy for printers
ErrorPolicy retry-job

# Only listen for connections from the local machine.
Listen localhost:631
Listen /run/cups/cups.sock

# Show shared printers on the local network.
Browsing Yes
BrowseLocalProtocols dnssd

# Default authentication type, when authentication is required...
DefaultAuthType Basic

# Web interface setting...
WebInterface Yes

# Timeout after cupsd exits if idle (applied only if cupsd runs on-demand - with -l)
IdleExitTimeout 60

# Restrict access to the server...
<Location />
  Order allow,deny
</Location>

# Restrict access to the admin pages...
<Location /admin>
  Order allow,deny
</Location>

# Restrict access to configuration files...
<Location /admin/conf>
  AuthType Default
  Require user @SYSTEM
  Order allow,deny
</Location>

# Restrict access to log files...
<Location /admin/log>
  AuthType Default
  Require user @SYSTEM
  Order allow,deny
</Location>

# Set the default printer/job policies...
<Policy default>
  # Job/subscription privacy...
  JobPrivateAccess default
  JobPrivateValues default
  SubscriptionPrivateAccess default
  SubscriptionPrivateValues default

  # Job-related operations must be done by the owner or an administrator...
  <Limit Create-Job Print-Job Print-URI Validate-Job>
    Order deny,allow
  </Limit>

  <Limit Send-Document Send-URI Hold-Job Release-Job Restart-Job Purge-Jobs Set-Job-Attributes Create-Job-Subscription Renew-Subscription Can
    Require user @OWNER @SYSTEM
    Order deny,allow
  </Limit>

  # All administration operations require an administrator to authenticate...
  <Limit CUPS-Add-Modify-Printer CUPS-Delete-Printer CUPS-Add-Modify-Class CUPS-Delete-Class CUPS-Set-Default CUPS-Get-Devices>
    AuthType Default
    Require user @SYSTEM
    Order deny,allow
  </Limit>

  # All printer operations require a printer operator to authenticate...
  <Limit Pause-Printer Resume-Printer Enable-Printer Disable-Printer Pause-Printer-After-Current-Job Hold-New-Jobs Release-Held-New-Jobs Deac
    AuthType Default
    Require user @SYSTEM
    Order deny,allow
  </Limit>

  # Only the owner or an administrator can cancel or authenticate a job...
  <Limit Cancel-Job CUPS-Authenticate-Job>
```

```
    Require user @OWNER @SYSTEM
    Order deny,allow
  </Limit>

  <Limit All>
    Order deny,allow
  </Limit>
</Policy>

# Set the authenticated printer/job policies...
<Policy authenticated>
  # Job/subscription privacy...
  JobPrivateAccess default
  JobPrivateValues default
  SubscriptionPrivateAccess default
  SubscriptionPrivateValues default

  # Job-related operations must be done by the owner or an administrator...
  <Limit Create-Job Print-Job Print-URI Validate-Job>
    AuthType Default
    Order deny,allow
  </Limit>

  <Limit Send-Document Send-URI Hold-Job Release-Job Restart-Job Purge-Jobs Set-Job-Attributes Create-Job-Subscription Renew-Subscription Can
    AuthType Default
    Require user @OWNER @SYSTEM
    Order deny,allow
  </Limit>

  # All administration operations require an administrator to authenticate...
  <Limit CUPS-Add-Modify-Printer CUPS-Delete-Printer CUPS-Add-Modify-Class CUPS-Delete-Class CUPS-Set-Default>
    AuthType Default
    Require user @SYSTEM
    Order deny,allow
  </Limit>

  # All printer operations require a printer operator to authenticate...
  <Limit Pause-Printer Resume-Printer Enable-Printer Disable-Printer Pause-Printer-After-Current-Job Hold-New-Jobs Release-Held-New-Jobs Deact
    AuthType Default
    Require user @SYSTEM
    Order deny,allow
  </Limit>

  # Only the owner or an administrator can cancel or authenticate a job...
  <Limit Cancel-Job CUPS-Authenticate-Job>
    AuthType Default
    Require user @OWNER @SYSTEM
    Order deny,allow
  </Limit>

  <Limit All>
    Order deny,allow
  </Limit>
</Policy>

# Set the kerberized printer/job policies...
<Policy kerberos>
  # Job/subscription privacy...
  JobPrivateAccess default
  JobPrivateValues default
  SubscriptionPrivateAccess default
  SubscriptionPrivateValues default

  # Job-related operations must be done by the owner or an administrator...
  <Limit Create-Job Print-Job Print-URI Validate-Job>
    AuthType Negotiate
    Order deny,allow
  </Limit>

  <Limit Send-Document Send-URI Hold-Job Release-Job Restart-Job Purge-Jobs Set-Job-Attributes Create-Job-Subscription Renew-Subscription Can
    AuthType Negotiate
    Require user @OWNER @SYSTEM
    Order deny,allow
  </Limit>

  # All administration operations require an administrator to authenticate...
  <Limit CUPS-Add-Modify-Printer CUPS-Delete-Printer CUPS-Add-Modify-Class CUPS-Delete-Class CUPS-Set-Default>
    AuthType Default
    Require user @SYSTEM
    Order deny,allow
  </Limit>
```

```
  # All printer operations require a printer operator to authenticate...
  <Limit Pause-Printer Resume-Printer Enable-Printer Disable-Printer Pause-Printer-After-Current-Job Hold-New-Jobs Release-Held-New-Jobs Deact
    AuthType Default
    Require user @SYSTEM
    Order deny,allow
  </Limit>

  # Only the owner or an administrator can cancel or authenticate a job...
  <Limit Cancel-Job CUPS-Authenticate-Job>
    AuthType Negotiate
    Require user @OWNER @SYSTEM
    Order deny,allow
  </Limit>

  <Limit All>
    Order deny,allow
  </Limit>
</Policy>
```

All the printer data is saved at `/etc/cups/printers.conf`. The web interface or any other GUI is actually editing this file.

Copy

```
# cat /etc/cups/printers.conf
# Printer configuration file for CUPS v2.4.2
# Written by cupsd
# DO NOT EDIT THIS FILE WHEN CUPSD IS RUNNING
NextPrinterId 2
<Printer MyPrinter>
PrinterId 1
UUID urn:uuid:cea56d60-93a0-31bf-443b-5a7288e2cd51
Info My Printer
Location other room
MakeModel HP DesignJet 600 pcl, 1.0
DeviceURI http://thatprinter:631/ipp/
State Idle
StateTime 1689596391
ConfigTime 1689596367
Type 8450116
Accepting Yes
Shared Yes
JobSheets none none
QuotaPeriod 0
PageLimit 0
KLimit 0
OpPolicy default
ErrorPolicy retry-job
</Printer>
```

Another important configuration directory is located at `/etc/cups/ppd/`. This directory contains the PostScript printer Description (PPD) files. These let printers which use them to function properly.

To find the CUPS logs, check the `/var/log` directory:

Copy

```
# ls /var/log/cups/
access_log  error_log
```

CUPS web interface

To enable CUPS's web interface, you have to enable the following configuration in the `/etc/cups/cupsd.conf`:

Copy

```
WebInterface Yes
```

Then you will have access to the GUI via port **631**. So it will be enough to access the **localhost:631** or **127.0.0.1:631** (or the server's **IP** address on port **631**) from your browser.

There are some of the important sections on the above page:

Section	Usage
Administration	Adding printers, managing jobs and configuring the CUPS server
Jobs	Checking the active, pending & completed jobs
Printers	List or search in the installed printers

By default, system users can view printers and queued jobs ,but changes (like adding printers) will need more access. This is configured at the bottom part of `cupsd.conf` file. For example, the below configuration provides `CUPS-Add-Modify-Printer CUPS-Delete-Printer CUPS-Set-Defaul` access to anyone in `@printer_admin` group.

Copy

```
<Limit CUPS-Add-Modify-Printer CUPS-Delete-Printer CUPS-Set-Default>
  AuthType Default
  Require user @printer_admin
  Order deny,allow
</Limit>
```

CUPS has most of the common printer drivers installed. You just need to choose the printer from the dropdown menu to add it.

legacy tools

Just like the MTA programs, CUPS support all the legacy command line programs too. These used to be printing commands in the BSD world so you may need to install the `cups-bsd` package to let them work under your CUPS environment. The table below lists the BSD printing compatibility commands:

command	usage
lpr	print a file
lpq	show print queue/jobs
lprm	rm/remove a file from priner queue
lpc	printer control / troubleshooting program

lpq

The **q** stands for **queue**. Therefor `lpq` shows the printer queue and is used when you want to see the printing jobs. If you use the `-a` switch, `lpq` will list the jobs from **all** printers. Alternatively you can use the `-P` switch to show the jobs of a specific printer. So the following command will show the jobs of a printer called Apple-Dot-Matrix:

Copy

```
# lpq -PApple-Dot-Matrix
Apple-Dot-Matrix is ready and printing
Rank    Owner   Job     File(s)                         Total Size
active  unknown 1       unknown                         7168 bytes
1st     unknown 2       unknown                         2048 bytes
```

There should be no spaces between the `-P` and the printer's name; strange? yes :D

lpr

This command sends a job to a printer. The printer is specified via the `-P` switch.

Copy

```
$ lpr -PApple-Dot-Matrix for_print.txt
$ lpq
Apple-Dot-Matrix is ready and printing
Rank    Owner   Job     File(s)                         Total Size
active  jadi    1       Untitled Document 1             7168 bytes
1st     jadi    2       Untitled1                       2048 bytes
2nd     jadi    3       for_print.txt                   1024 bytes
```

If no printer is specified, the default printer will be used

lprm

The **rm** stands for **remove**. Therefor the `lprm` command removes jobs from the queue. Obviously, you have to provide the **Job ID**.

Copy

```
$ lpq
Apple-Dot-Matrix is ready and printing
Rank    Owner   Job     File(s)                         Total Size
active  jadi    1       Untitled Document 1             7168 bytes
1st     jadi    2       Untitled1                       2048 bytes
2nd     jadi    3       for_print.txt                   1024 bytes
jadi@funlife:/tmp$ lprm 2
jadi@funlife:/tmp$ lpq
Apple-Dot-Matrix is ready and printing
Rank    Owner   Job     File(s)                         Total Size
active  jadi    1       Untitled Document 1             7168 bytes
1st     jadi    3       for_print.txt                   1024 bytes
```

Only root can remove other peoples print jobs

If you need to remove ALL the jobs of a specific printer, you can go with `-Pprinter_name -`. Yes! That is only one dash (`-`) after the printer name; that's why this is called a legacy command.

the `lprm -` will remove all the print jobs

lpc

The **c** stands for **control**. Therefor `lpc` lets you control, check the status (via `lpc status`) and troubleshoot your printers.

Copy

```
$ lpc status
Apple-Dot-Matrix:
    printer is on device 'ipp' speed -1
    queuing is enabled
    printing is enabled
    2 entries
    daemon present
```

In the above response,

- **queuing is enabled** tell us that the queue can accept new print jobs. If the queue is disabled, you can not even send new jobs to the printer.
- **printing is enabled** means that the printer is actually can print on the paper. This will be on the disable state if the printer is out of ink or paper or experiencing a paper jam.

If you are having problems with your printer or need to prevent it from accepting new jobs or let it accept jobs but not do the actual printing, these four commands can help:

command	usage
cupsaccept	tells the printer queue to accept new jobs
cupsreject	tells the printer to reject any new job
cupsenable	enables the actual/physical printing of the jobs
cupsdisable	disables the physical printing of the jobs

In all cases, you have to provide the printer name of the printer. It is also possible to provide a reason using `-r` switch.

Copy

```
$ cupsdisable Apple-Dot-Matrix -r "need more paper"
$ lpc status
Apple-Dot-Matrix:
    printer is on device 'ipp' speed -1
    queuing is enabled
    printing is disabled
    2 entries
    daemon present
```

109.1 Fundamentals of internet protocols

Weight: 4

Candidates should demonstrate a proper understanding of TCP/IP network fundamentals.

Key Knowledge Areas

- Demonstrate an understanding of network masks and CIDR notation.
- Knowledge of the differences between private and public "dotted quad" IP addresses.
- Knowledge about common TCP and UDP ports and services (20, 21, 22, 23, 25, 53, 80, 110, 123, 139, 143, 161, 162, 389, 443, 465, 514, 636, 993, 995).
- Knowledge about the differences and major features of UDP, TCP and ICMP.
- Knowledge of the major differences between IPv4 and IPv6.
- Knowledge of the basic features of IPv6.

Terms and Utilities

- /etc/services
- IPv4, IPv6
- Subnetting
- TCP, UDP, ICMP

TCP/IP

The *Transmission Control Protocol/Internet Protocol* (or TCP/IP in short) is stack or protocols which powers the most of communications on the Internet (and many other networks). Although it is generally called TCP, it also includes UDP, ICMP, DNS and many others.

IP

Internet Protocol or IP is a way to logically pointing to a host on the Internet. When a device has an IP Address, it is possible to send data toward it. The IP addresses should be unique so the network will know where each packet is intended to go. There are 2 active versions of the IP protocol; 4 & 6.

An IPv4 is in the form of A.B.C.D where A, B, C & D are between 0-255. The followings are all valid IP addresses:

Copy

```
1.2.3.4
1.1.1.1
100.0.0.100
192.168.1.4
```

> each part of an IP address is called an Octet since it is constructed of an 8 bit number (0..255) and can be shown in binary or hex or any notation, but Decimal format is more common.

Having in mind that each octet can be between 0 to 255, in total we can have 256 * 256 * 256 * 256 = 4294967296 IPv4 addresses; That is around 4.3 billion - only! Thats why we are switching to IPv6.

Private IPs

IP addresses should be uniq on a network so people should get their IP addresses from a central authority called Internet Assigned Numbers Authority (IANA). The available IP addresses are grouped in 3 classes:

Class	First Octet	Range
A	1-126	`1.0.0.0` to `126.255.255.255`
B	128-191	`128.0.0.0` to `191.255.255.255`
C	192-223	`192.0.0.0` to `223.255.255.255`

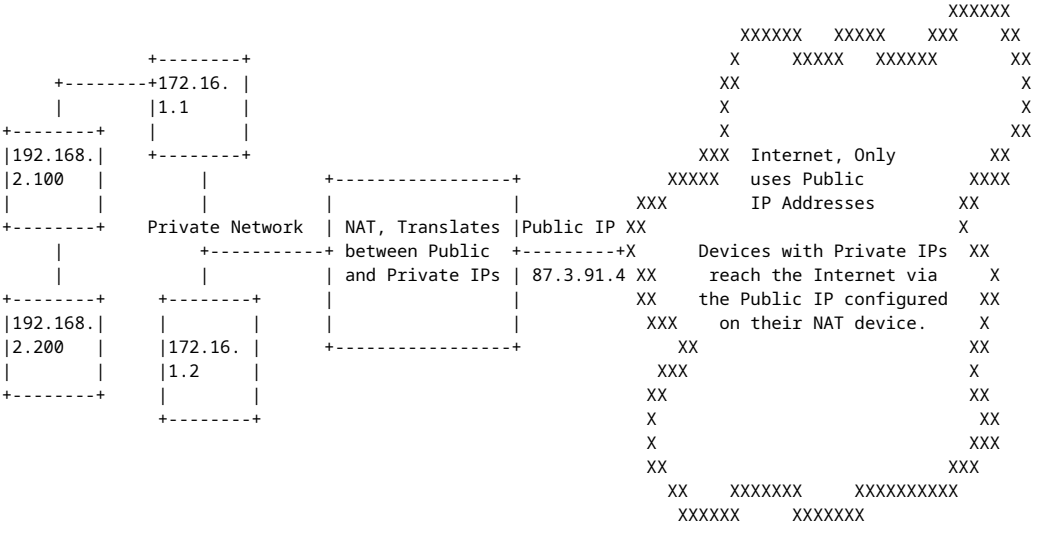
Normally if you are creating a server on your service provider, that service provider will give you an IP address purchased from IANA (via smaller regional registries). But what happens if you want to assign an IP to your laptop or mobile phone of tablet *inside* your home? These devices are not visible directly on the Internet and you are not going to purchase IPs for them.

Here the *Private IP Address* ranges come to play. The below ranges are called "private" addresses and anyone can assign them to any device they want, as long as they are not visible on the open internet.

range	Number of IPs in this range
192.168.0-255.0-255	65K IPs
172.16-31.0-255.0-255	1M IPs
10.0-255.0-255.0-255	16M IPs

Any one can use any of these IPs on her devices as long as it is not connected directly to the internet. Have a look at this example:

Copy



In the middle you can see a `Network Address Translation` or `NAT` device. It is connected to the Internet using only one public IP (It might be 87.3.91.4). On the left side of the NAT box, there are 4 devices with Private IPs. Each time any of these devices targets an address on the Internet (a public IP), the NAT device will use its own public IP to request that IP and when the answer is ready, will provide it to the *Natted* device - the device behind the NAT. This way we can create huge networks behind a NAT device. In this scenario, all the devices can reach the Internet but no-one from the Internet can directly reach any of them, directly; Unless we've configured an DMZ.

Subnetting (and Netmask)

OK! We have around 4B addresses in IPv4. But what happens if I send a packet to another IP? Say I try to print a document on the *local* printer using its private IP as you learned in previous section? Who should listen for this packet? Does the whole Internet passes it hand to hand? Obviously not! Only my *local* network should be able to receive that packet and in this specific case, only they printer should receive it on its local IP.

This is achieved by subnetting. The purpose of subnetting is to create *sub-networks* in your IP address range. You have already seen this times and times when talking about your Netmask (the 255.255.255.0 is a famous one) or its CIDR notation which is `/24`. Lets have a deeper look.

> CIDR stands for Classless Inter-Domain Routing

When speaking about networks, you have to decide on

1. How many bits of my IP address is going to be shared between all my devices (network)
2. How many bits of my IP address is going to differentiate my devices in this network (hosts)

For example if you assign the `192.168.1.1` to your router and set the Netmask on `/24`, you are claiming 24 (left hand bits) of `192.168.1.1` as your Network (so `192.168.1.` because 245 = 8 * 3) and 8 bits as your Host (last octet). In this network you can assign these IPs to your devices: `192.168.1.10`, `192.168.1.2`, `192.168.1.200` and they will see each other correctly.

There are 2 methods to talk about Netmask bits. The CIDR `/24` method and Decimal Netmask (255.255.255.0) method. In CIDR we are saying how many bits are 1 (the network section of an IP address) and in Decimal method, we are representing these 1 bits in an IP address format. Have a look at the following table:

Decimal	CIDR	Binary
255.0.0.0	/8	11111111.00000000.00000000.00000000
255.255.0.0	/16	11111111.11111111.00000000.00000000
255.255.255.0	/24	11111111.11111111.11111111.00000000

> In above table I'm just showing the easy ones ;) If you are taking an CCNA exam, you should be able to calculate the Network/Host section for something like /13; which is not difficult now that you know the concept.

Any time any packets tries to reach any IP in my **subnet** (that is 192.168.1.0-255), my router routes the packets correctly to the destination. But If any devices tries to reach anything out of my subnet (say a public IP), my router will use its NAT functionality (or other methods) to send the packets over the Internet (or the correct network based on the routing table).

> for a better understanding of subnetting, have a look at Cisco document

Network and Broadcast Address

When having the IP address and the Network Mask, we can calculate the network address and the broadcast address.

The network address is a Logical AND between the IP address and the networkmask. The broadcast address is the network address with a Logical OR when all the host bits are changed to 1.

Confusing... yes. Because talking about maths is difficult. Say we have the `192.168.4.12` as our IP Address and our CIDR is /24 (that is 255.255.255.0). Lets calculate the network and broadcast addresses. `ipcalc` a useful utility you can use.

Copy

```
IP:        11000000.10101000.00000100.00001100 (192.168.4.12)
Netmask:   11111111.11111111.11111111.00000000 (255.255.255.0)
Network:   11000000.10101000.00000100.00000000 (192.168.4.0)
Broadcast: 11000000.10101000.00000100.11111111 (192.168.4.255)
```

Binary to Decimal Conversion

You are probably familiar with binary base - it is the basis of anything related to computers. If not I highly recommend you to do a google search and study it from another source since this book assumes you already know it. In short when writing numbers on binary format (base 2), the only used digits are 0 and 1. An easy way for conversion is using the following table to convert 0 and 1s to our *normal* decimal format. Its enough to know the value of each bit and add up the ones having 1 in them.

128	64	32	16	8	4	2	1	Decimal
0	0	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0	128
1	0	0	0	0	0	0	1	129
0	0	1	0	0	0	0	0	32
1	0	0	0	0	1	0	0	132
0	0	0	0	0	1	1	0	6
1	0	1	1	0	0	1	1	179
1	1	1	1	1	1	1	1	255

So `11000000.10101000.00000001.00001111` equals to `192.168.1.15`.

Communication Protocols

When computers communicate, they use Protocols. This protocols are designed to let computers speak in different ways for fulfilling different needs. Here we will have a look at three of the most popular protocols on the Internet: TCP / UDP / ICMP.

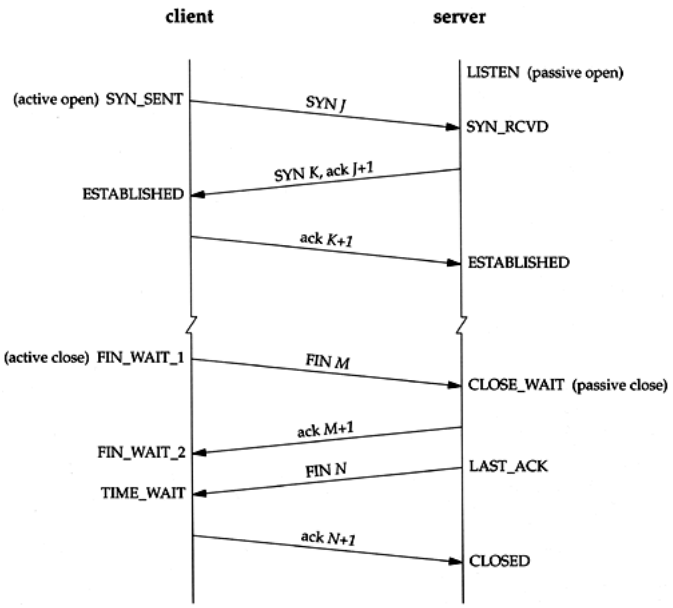
TCP

Transmission Control Protocol or TCP is designed to make sure that both parties are *speaking* with each other without loosing any information. In this protocol the receiver will get anything the sender sends, exactly as it is sent. When you are downloading a program from the Internet, TCP is the best option because you need the file to be downloaded **exactly** as it is stored on the server. The TCP communication is kind of like this:

Copy

```
A- Are you ready?
B- Yes
A- Please share with me the file XYZ
B- Ok. Are you ready?
A- Yes. Start sending
B- OK. This is the part 1 of XYZ
A- OK I got it based on these criteria?
B- Good. Now is it correct?
A- Yes. Please send the second part
B- OK. Are you ready for the second part?
```

```
A- Yes.
B- Here comes the second part
A- ...
B- ...
```



As you can see, TCP needs a lot of communications and spends time (or even retransmits data) to make sure that the receiver is getting the exact correct data available on the server. Strangely in some cases this is not what we are looking for. If you are watching a movie or having a phone call or enjoying a game stream, you prefer to continue watching live in case of a 1s issue in your network. Thats why we also have the UDP.

UDP

You are video-chatting with a friend and network fluctuates. What is a better choice?
A) retransmitting the missing packets and/or reestablishing the connection and continue the whole conversation with a 2s delay or ...
B) just show the newer packets we got and continue the live vide-conference and just forget about that 2 second fluctuation (missed data)? If your choice is B, it is better if you use UDP (User Datagram Protocol) for your chat program. UDP is less reliable: the sender sends packets without communicating much and hearing back from the receiver and receiver listens for packets without negotiating the exact details with the sender. It is much faster than TCP but you can not be sure that 100% of packets will be received by the B party.

ICMP

Internet Control Messaging Protocol or ICMP is a specific purpose protocol used to check the connectivity of the servers by the `ping` command. The first computer just tells "are you there?" and the second will answer "yes I'm here". Have a look at this practical example:

Copy

```
[jadi@funlife ~]$ ping google.com
PING google.com (173.194.32.135) 56(84) bytes of data.
64 bytes from 173.194.32.135: icmp_seq=1 ttl=48 time=239 ms
64 bytes from 173.194.32.135: icmp_seq=2 ttl=48 time=236 ms
^C
--- google.com ping statistics ---
2 packets transmitted, 2 received, 0% packet loss, time 1001ms
rtt min/avg/max/mdev = 236.707/238.174/239.642/1.546 ms
```

As you can see, we are pinging (send ICMP packets) to a server and it answers are back (replying to our ICMP packets).

> ICMP is used for network troubleshooting and DOES NOT transfers user data

Port

Any computer has an address but there are many programs running on that computer. By using an IP address you can tell the Internet the destination of your packets but how can you decide which program on that computer should answer to your packet? We know that a computer with the address of 5.1.23.1 has a webserver and a ftp server on it but how can we reach it and tell it "i want the index.html of your webserver" or "deliver me the XYZ file from your FTP server"? This is done using PORTs. Ports are numbers where a program LISTENS to. For example port 80 is reserved for webservers so if I connect to 5.1.23.1:80 I'm sure that I'm talking with the webserver. In the same way, when the file transport protocol (FTP) starts, it starts listening on port 20 & 21 and if I use a FTP client to reach that computer, I will automatically connect to port 21 which is reserved for FTP.

Different ports can use different transmission protocols (UDP or TCP). The default port of some protocols are as follow. These are very important and most admins know them.

port	usage
20, 21	FTP (one data, one control)
22	SSH
23	Telnet
25	SMTP
53	DNS
80	HTTP
110	POP3
123	NTP
139	NetBIOS
143	IMAP
161, 162	SNMP
389	LDAP
443	HTTPS
465	SMTPS
636	LDAPS
993	IMAPS
995	POP3S

Note that in this table all ports above 400 ends with S, which stands for **S**ecure

You can find all of the above ports and many many others in `/etc/services`

IPv6

We saw that there are only around 4B IPv4s available. Just consider that we are around 7B people on the planet earth so there is not even enough IP for every person on IPv4 range. Add to this the latest demands from all the mobile phones, cars, fridges, clocks, TVs, camera, ...! They all want to be on the Internet; this is called Internet Of Things. What should be done? We saw one solution called NAT but the permanent solution is a new version of the IP; IP version 6. In version 6 IPs are not limited to 4 octets anymore.

Each IPv6 has 128 bits. They are grouped in 8, 16 bit groups and are normally written in Hex (base 16) format. A sample IPv6 address looks like `2001:0db8:0a0b:12f0:0000:0000:0000:0001` and it can be shortened to `2001:db8:a0b:12f0::1`. Here we have decimal numbers from **0000** to **FFFF** and **8** fields which are separated by **:**.

IPv6 provides us around 3.4*(10^38) IPs which is enough for whatever we can imagine at the moment. Just imagine that it can allocate 2^52 addresses for every observable star in the known universe.

109.2 Persistent network configuration

Weight: 4

Candidates should be able to view, change and verify configuration settings on client hosts.

Key Knowledge Areas

- Understand basic TCP/IP host configuration
- Configure ethernet and wi-fi network configuration using NetworkManager
- Awareness of systemd-networkd

Terms and Utilities

- `/etc/hostname`
- `/etc/hosts`
- `/etc/nsswitch.conf`
- `/etc/resolv.conf`
- `nmcli`
- `hostnamectl`
- `ifup`

- `ifdown`

Intro

As we saw in the previous section, every PC, server, laptop, phone, .. should have an IP configuration (IP, Netmask, Default gateway, DNS, ..) to work properly in the network. This can be done in various ways. Some devices like laptops are changing their network all the time and should be able to keep up with the changes. Some servers remain in the same location (physical and network wise) all their life and should persist this configuration after restarts, outages, upgrade and HW changes.

In this section we will see how this can be achieved in modern GNU/Linux systems.

Network Interface

The NIC (or Network Interface Card) is the physical network hardware in your computer. This can be the chip+antenna in your mobile phone or an Ethernet Card connected to a network cable on your PC.

In older systems, these were called things like `eth0`, `eth1`, `eth2`, ... where 0, 1 & 2 were decided by the kernel - mostly based on the order of loading the drivers. In recent Linux machines the NICs are called by `wlan0`, `eno1`, `ens1`, `enp3s2` and such. This is based on some more concrete data like being an `wireless` or `ethernet`, PCI (`ens`) or bus like (`enp`).

The `ip` command can show these:

Copy

```
➜  ~ ip link show
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN mode DEFAULT group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
2: wlp108s0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc noqueue state UP mode DORMANT group default qlen 1000
    link/ether 00:bb:60:97:6b:07 brd ff:ff:ff:ff:ff:ff
3: docker0: <NO-CARRIER,BROADCAST,MULTICAST,UP> mtu 1500 qdisc noqueue state DOWN mode DEFAULT group default
    link/ether 02:42:75:d3:e6:ff brd ff:ff:ff:ff:ff:ff
```

> The `lo` is a virtual network adapter called the *loopback* device. It is always there and points to "this device or 127.0.0.1 as IPv4 calls it".

Configuring NICs

In older distributions, the `ifconfig` was used to check / configure the IP settings on NICs. Have a look:

Copy

```
$ ifconfig
enp0s25: flags=4099<UP,BROADCAST,MULTICAST>  mtu 1500
        ether f0:de:f1:62:c5:73  txqueuelen 1000  (Ethernet)
        RX packets 0  bytes 0 (0.0 B)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 0  bytes 0 (0.0 B)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
        device interrupt 20  memory 0xd1500000-d1520000

lo: flags=73<UP,LOOPBACK,RUNNING>  mtu 65536
        inet 127.0.0.1  netmask 255.0.0.0
        inet6 ::1  prefixlen 128  scopeid 0x10<host>
        loop  txqueuelen 1  (Local Loopback)
        RX packets 560719  bytes 339937974 (324.1 MiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 560719  bytes 339937974 (324.1 MiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

wlp3s0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 192.168.1.35  netmask 255.255.255.0  broadcast 192.168.1.255
        inet6 fe80::8ea9:82ff:fe7b:8906  prefixlen 64  scopeid 0x20<link>
        ether 8c:a9:82:7b:89:06  txqueuelen 1000  (Ethernet)
        RX packets 2325385  bytes 2629859900 (2.4 GiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 2023796  bytes 510997240 (487.3 MiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

It is also possible to use `ifconfig` to change the network configurations, but you should have root access:

Copy

```
$  sudo ifconfig enp0s25 192.168.42.42
password for jadi:
$ ifconfig enp0s25
enp0s25: flags=4099<UP,BROADCAST,MULTICAST>  mtu 1500
        inet 192.168.42.42  netmask 255.255.255.0  broadcast 192.168.42.255
        ether f0:de:f1:62:c5:73  txqueuelen 1000  (Ethernet)
        RX packets 0  bytes 0 (0.0 B)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 0  bytes 0 (0.0 B)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
        device interrupt 20  memory 0xd1500000-d1520000

$
```

In case you want to change the netmask of an interface, do `ifconfig eth0 netmask 255.255.0.0` or as most of us used to do, issue both in one command:

Copy

```
# ifconfig eth0 192.168.42.42 netmask 255.255.255.0
```

It is also possible to turn the interfaces *up* and *down* (on and off) using predefined configurations by:

Copy

```
$ sudo ifconfig enp0s25 down
[sudo] password for jadi:
$ ifconfig
lo: flags=73<UP,LOOPBACK,RUNNING>  mtu 65536
        inet 127.0.0.1  netmask 255.0.0.0
        inet6 ::1  prefixlen 128  scopeid 0x10<host>
        loop  txqueuelen 1  (Local Loopback)
        RX packets 562273  bytes 340257228 (324.4 MiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 562273  bytes 340257228 (324.4 MiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

wlp3s0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 192.168.1.35  netmask 255.255.255.0  broadcast 192.168.1.255
        inet6 fe80::8ea9:82ff:fe7b:8906  prefixlen 64  scopeid 0x20<link>
        ether 8c:a9:82:7b:89:06  txqueuelen 1000  (Ethernet)
        RX packets 2330388  bytes 2634026235 (2.4 GiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 2027352  bytes 511549072 (487.8 MiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

As you can see, downing the interface removed it from the list of active interfaces, using switch `-a` will tell the `ifconfig` to show ALL interfaces, even if they are down.

In many systems there are `ifup` and `ifdown` commands directly to up and down interfaces easily. They work just like `ifup eth0`.

These predefined configs are located at `/etc/network/interfaces` on Debian based machines and at `/etc/sysconfig/network-scripts/` in RPM based distro.

This is a sample of such file on a RedHat based distro:

Copy

```
$ cat /etc/sysconfig/network-scripts/ifcfg-eth0
DEVICE=eth0
ONBOOT=yes
TYPE=Ethernet
IPADDR=192.168.1.10
NETMASK=255.255.255.0
DNS1=4.2.2.4
```

On RPM systems, the default gateway is configured via the below file:

Copy

```
cat /etc/sysconfig/network
NETWORKING=yes
HOSTNAME=lpictest
GATEWAY=192.168.1.1
```

On Debian based systems (including Ubuntu) the main configuration file for network interfaces is `/etc/network/interfaces`. This one file has the configuration for all the interfaces. Have a look:

Copy

```
auto lo
iface lo inte loopback

auto eth0
#ifconfig eth0 inet dhcp
iface eth0 inet static
address 192.168.1.10
netmask 255.255.255.0
gateway 192.168.1.1
dns-nameservers 4.2.2.4
```

ip

Recent distributions are mostly using the `ip` command. This command can do lots of things including and not limited to showing and configuring the IP addresses, netmasks, default gateways & routing rules.

Copy

```
ip addr add 172.19.1.10/24 dev eth2 # temporary adding an IP
ip addr show eth2
ip addr del 172.19.1.10/24 dev eth2 # deleting an IP address
ip link set eth2 up # brining a NIC up
ip route show
ip route add default via 192.168.1.1 # add default gateway
```

Please note that above commands do temporary changes which will be lost after restarting the `NetworkManager` service. If you need permanent changes, it should be done using the configuration files or Network Manager interfaces.

NetworkManager & `nmcli`

In recent years, `NetworkManager` services has gained a lot of popularity. This service can "watch" the status of network and various configuration and configure the network cards (specially the wifi ones) accordingly. This is what makes our laptop connected whenever we open it in an area with a known WiFi or ask about the password if we want to connect to a new network or assign IP addresses as soon as we connect the cable to our Ethernet card. This IP assignment might happen via the "permanent IP configuration" on your device or a protocol called DHCP. When using DHCP (Dynamic Host Configuration Protocol), your computer asks a DHCP server (say your home's wifi router) about the IP, Netmask, Default gateway, DNS and other stuff and sets them.

By default, NetworkManager daemon controls the networks which are not mentioned in `/etc/network/interfaces`. This service is running in the background and controls the NICs which are not configured there. Various frontend GUI (graphical user interface) or TUI (textual user interface. try `nmtui`) or CLI (command line interfaces) programs exists to control or configure the NetworkManager daemon. If you are using a Desktop Linux, you've probably already used / know one (say the network manager applet). Here I will show you how to use the `nmcli` from the command line.

We always call the `nmcli` with one of it various commands, here is a list:

Command	Usage
general	NetworkManager’s general status and operations.
networking	Overall networking control.
radio	NetworkManager radio switches.
connection	Controlling the connection
device	Devices controlled by NetworkManager
agent	secret or polkit agent
monitor	Montor the changes

For example, we can check the current status with the `general` command:

Copy

```
➜  ~ nmcli general
STATE      CONNECTIVITY  WIFI-HW  WIFI     WWAN-HW  WWAN
connected  full          enabled  enabled  missing  enabled
```

Or if you want to check the devices or list of Wi-Fi connections:

Copy

```
➜  ~ nmcli device
DEVICE            TYPE      STATE                   CONNECTION
wlp108s0          wifi      connected               Sharm Bar Sansoor 5
docker0           bridge    connected (externally)  docker0
lo                loopback  connected (externally)  lo
p2p-dev-wlp108s0  wifi-p2p  disconnected            --
➜  ~ nmcli device wifi
IN-USE  BSSID              SSID                 MODE   CHAN  RATE        SIGNAL  BARS  SECURITY
        6C:AD:EF:38:13:38  AxLTE                Infra  3     270 Mbit/s  84      ▂▄▆█  WPA2
        00:E0:4C:93:1D:B8  Lanat Be Sansoorchi  Infra  6     130 Mbit/s  59      ▂▄▆_  WPA2
*       24:F5:A2:42:DE:CE  Sharm Bar Sansoor 5  Infra  36    540 Mbit/s  47      ▂▄__  WPA2
        30:A2:20:DD:8B:54  AvinaAmin            Infra  7     270 Mbit/s  29      ▂___  WPA1 WPA2
        30:85:A9:8C:71:2C  bahram               Infra  11    65 Mbit/s   29      ▂___  WPA2
```

To connect to a Wi-Fi network, you're a do:

Copy

```
nmcli device wifi connect AxLTE password AFunkyPassword
```

Fancy Names for Computers

hostname

Remembering IP addresses are easy for robots but not for humans. Thats why we have "hostname"s. A hostname is a like a contact list where you just tell "call Jadi" and the system known my phone number. If you check your `/etc/hostname`, you will see your machines name there. Although you can change it temporarily (or permanently). The command is `hostnamectl`.

Copy

```
[funlap ~]# hostnamectl set-hostname mycoolmachine
[funlap ~]# hostname
mycoolmachine
[funlap ~]# cat /etc/hostname
mycoolmachine
[funlap ~]# bash
[mycoolmachine ~]#
```

Or you can change it as *transient*, which is a temporary change using the `--transient` switch.

It is also possible to define a "pretty" name for your computer so other systems might show it in their interfaces more nicely:

Copy

```
[mycoolmachine ~]# hostnamectl --pretty set-hostname "LAN Shared Storage"
[mycoolmachine ~]# hostnamectl status
 Static hostname: mycoolmachine
 Pretty hostname: LAN Shared Storage
       Icon name: computer-convertible
         Chassis: convertible
      Machine ID: 0b126c4b6f4347168140eaa6202ce8be
         Boot ID: 675eff37f42648c6bdea31177596557f
Operating System: Manjaro Linux
          Kernel: Linux 6.1.38-1-MANJARO
    Architecture: x86-64
 Hardware Vendor: Dell Inc.
  Hardware Model: Latitude 7390 2-in-1
Firmware Version: 1.30.0
```

`/etc/hosts`

This file contains a list of IPs and their corresponding names, including your own computers.

Copy

```
[mycoolmachine ~]# head -20 /etc/hosts
##
# Host Database
#
# localhost is used to configure the loopback interface
# when the system is booting.  Do not change this entry.
##
127.0.0.1   localhost funlife db
255.255.255.255 broadcasthost
::1             localhost
##
# Host Database
#
# localhost is used to configure the loopback interface
# when the system is booting.  Do not change this entry.
```

```
##

198.74.56.50 jobs.jadi.net
192.168.1.22 amoledtesting

67.217.170.72 vps
```

So when you need to reach a machine by its name, your OS will now which IP to reach.

DNS configuration

DNS (which stands for Domain Name System) is a server which translates human-readable domain names (or more technically, text based domain names) to the corresponding IP addresses. You have to configure your computer to use a DNS so it will know which IP to contacted if you wanted to reach `linux1st.com` (and donate maybe).

This configuration can be found in `/etc/resolve.conf`.

Copy

```
nameserver 192.168.1.1
nameserver 4.2.2.4
domain jadi.net
search jadi.net company.com
```

Here I'm telling my computer to contact the DNS on my home network (192.168.1.1) or a DNS located at 4.2.2.4 if it needed to translate an address to an IP.

The `domain` configuration sets a local domain name so the machines in this domain will be able to use a short name (tv, instead of tv.jadi.net) and the `search` config does kind the same and tells the resolver to search for `tv.jadi.net` and `tv.company.com` if it was trying to resolve `tv`.

nsswitch

The `/etc/nsswitch.conf` file is used to configure which services are to be used to determine information such as hostnames, password files, and group files. Mine is

Copy

```
# cat /etc/nsswitch.conf
# Begin /etc/nsswitch.conf

passwd: files
group: files
shadow: files

publickey: files

hosts: files dns myhostname
networks: files

protocols: files
services: files
ethers: files
rpc: files

netgroup: files

# End /etc/nsswitch.conf
```

So if someone wants to check a password, the system will try the password *file* on the system. Or if they want to check an ip address of a hostname, my config says `hosts: files dns myhostname` so the computer first tries the files (/etc/hosts) and then goes for DNS. If I reverse these and change the line to

Copy

```
hosts:      dns files
```

Any resolve request will be sent to a DNS server first and the /etc/hosts will be used only if the DNS servers answers "I don't know!"

109.3 Basic network troubleshooting

Weight: 4

Candidates should be able to troubleshoot networking issues on client hosts.

Key Knowledge Areas

- Manually configure network interfaces, including viewing and changing the configuration of network interfaces using iproute2.
- Manually configure routing, including viewing and changing routing tables and setting the default route using iproute2.
- Debug problems associated with the network configuration.
- Awareness of legacy `net-tools` commands.

Terms and Utilities

- `ip`
- `hostname`
- `ss`
- `ping`
- `ping6`
- `traceroute`
- `traceroute6`
- `tracepath`
- `tracepath6`
- `netcat`
- `ifconifg`
- `netstat`
- `route`

Troubleshooting network problems

When a network related problem is reported to you, you have to take a lot of steps to find out where the root cause of the problem is. For example, if the report says "I can not open webpages", you have to start from checking if the network interface has an ip, if it is up, its the routing is OK and if the DNS works fine and if everything is OK, you have to try reaching a server on the Internet via `ping` command and if you see any problems, you have to use `traceroute` to see where your traffic is going wrong. In this lesson, we will review these basic steps.

ifconfig & ip

As you already know, `ifconfig` and `ip` commands can be used to check the IP address of your interfaces. If a network card is going to work, it needs a correct IP address and netmask. Let me check my own computer to see it has an IP address:

Copy

```
[jadi@debian ~]$ ip addr show
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
       valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
       valid_lft forever preferred_lft forever
2: wlp3s0: <NO-CARRIER,BROADCAST,MULTICAST,UP> mtu 1500 qdisc pfifo_fast state DOWN group default qlen 1000
    link/ether f0:de:f1:62:c5:73 brd ff:ff:ff:ff:ff:ff
3: enp0s25: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc mq state UP group default qlen 1000
    link/ether 8c:a9:82:7b:89:06 brd ff:ff:ff:ff:ff:ff
    inet 192.168.1.35/24 brd 192.168.1.255 scope global dynamic wlp3s0
       valid_lft 254836sec preferred_lft 254836sec
    inet6 fe80::8ea9:82ff:fe7b:8906/64 scope link
```

```
       valid_lft forever preferred_lft forever
[jadi@debian ~]$
[jadi@debian ~]$
[jadi@debian ~]$
[jadi@debian ~]$ ifconfig
enp0s25   Link encap:Ethernet  HWaddr f0:de:f1:62:c5:73
          inet addr:192.168.1.35  Bcast:192.168.1.255  Mask:255.255.255.0
          inet6 addr: fe80::8ea9:82ff:fe7b:8906/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:231586 errors:0 dropped:0 overruns:0 frame:0
          TX packets:200220 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:198053888 (198.0 MB)  TX bytes:51583154 (51.5 MB)

lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          inet6 addr: ::1/128 Scope:Host
          UP LOOPBACK RUNNING  MTU:65536  Metric:1
          RX packets:221752 errors:0 dropped:0 overruns:0 frame:0
          TX packets:221752 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:150859909 (150.8 MB)  TX bytes:150859909 (150.8 MB)
```

Both commands show that my IP address is OK. This is assigned to may be the Wi-Fi modem and 192.168.1.35 as IP and 255.255.255.0 looks reasonable.

> Please note that you can get a full help on `ip` using the `man ip` or get manual of a specific section using the `man ip-address` (or any other subcommand)

ping & ping6

This is the most common tool when troubleshooting a network problem. It sends an ICMP packet to a destination and if it gets back an answer, will inform you about not only it, but all the stats. Below, I will check my default route and will try to ping it. You should always be able to ping your default router, although sometimes paranoid sysadmins block the ICMP on the network and although you are connected, you won't get back answers.

Copy

```
jadi@debian:~$ ip route show
default via 192.168.70.1 dev enp0s1
172.17.0.0/16 dev docker0 proto kernel scope link src 172.17.0.1 linkdown
192.168.70.0/24 dev enp0s1 proto kernel scope link src 192.168.70.2
jadi@debian:~$ ping 192.168.70.1
PING 192.168.70.1 (192.168.70.1) 56(84) bytes of data.
64 bytes from 192.168.70.1: icmp_seq=1 ttl=64 time=1.11 ms
64 bytes from 192.168.70.1: icmp_seq=2 ttl=64 time=0.855 ms
^C
--- 192.168.70.1 ping statistics ---
2 packets transmitted, 2 received, 0% packet loss, time 1002ms
rtt min/avg/max/mdev = 0.855/0.984/1.113/0.129 msec
```

I can ping my gateway, but is it possible to reach a server on the Internet using its IP address. To find the answer, let's ping 4.2.2.4; it is very well known server on the Internet and many people use it to check their IP connectivity.

Copy

```
[jadi@debian ~]$ ping 4.2.2.4
PING 4.2.2.4 (4.2.2.4) 56(84) bytes of data.
64 bytes from 4.2.2.4: icmp_seq=1 ttl=50 time=108 ms
64 bytes from 4.2.2.4: icmp_seq=2 ttl=50 time=111 ms
64 bytes from 4.2.2.4: icmp_seq=3 ttl=50 time=113 ms
^C
--- 4.2.2.4 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2003ms
rtt min/avg/max/mdev = 108.160/111.233/113.717/2.338 ms
```

This is also working fine. But can I ping google.com too?

Copy

```
[jadi@debian ~]$ ping google.com
ping: unknown host google.com
```

Aha! We found the problem. In this case I have a correct IP address on my machine, I can ping my default gateway and I can ping 4.2.2.4, but I can not ping google.com. The error message is "unknown host". This means my computer can not translate google.com to its IP address; this is a DNS issue:

Copy

```
[jadi@debian ~]$ cat /etc/resolv.conf
# Dynamic resolv.conf(5) file for glibc resolver(3) generated by resolvconf(8)
#     DO NOT EDIT THIS FILE BY HAND -- YOUR CHANGES WILL BE OVERWRITTEN
```

No wonder, we had problems with surfing the WWW. There is no active DNS in my computer so no one will be able to reach sites by their domain names! This should be fixed by adding a DNS to my configuration file.

routing problems

In some situations you can not reach the Internet (say 4.2.2.4 or 8.8.8.8) but you can ping your gateway. Have a look:

Copy

```
[jadi@debian ~]$ ping 8.8.8.8
connect: Network is unreachable
[jadi@debian ~]$ ping 192.168.1.1
PING 192.168.1.1 (192.168.1.1) 56(84) bytes of data.
64 bytes from 192.168.1.1: icmp_seq=1 ttl=254 time=3.03 ms
64 bytes from 192.168.1.1: icmp_seq=2 ttl=254 time=3.31 ms
^C
--- 192.168.1.1 ping statistics ---
2 packets transmitted, 2 received, 0% packet loss, time 1001ms
rtt min/avg/max/mdev = 3.039/3.174/3.310/0.146 ms
```

What is going on here? Let's see, what clues do I have: 1) I can reach the gateway. 2) when asking for the Internet, my computer does not know what to do. In this case the **default gateway** is missing: the computer does not know what to do if a packet is outside its network mask. You know that we can set the default gateway using the `/etc/network/interfaces` config file, but there is also a `route` (or the newer `ip route` subcommand) to show and change the routing configurations on the fly.

> Routes added or changed via `ip route` (or `route`) will be lost after a reboot! Permanent configurations should come from configuration files.

Lets check our current routing state using `route` command as root.

Copy

```
jadi@debian:~$ ip route show
default via 192.168.70.1 dev enp0s1
172.17.0.0/16 dev docker0 proto kernel scope link src 172.17.0.1 linkdown
192.168.70.0/24 dev enp0s1 proto kernel scope link src 192.168.70.2

jadi@debian:~$ sudo ip route del default
[sudo] password for jadi:

jadi@debian:~$ ip route show
172.17.0.0/16 dev docker0 proto kernel scope link src 172.17.0.1 linkdown
192.168.70.0/24 dev enp0s1 proto kernel scope link src 192.168.70.2

jadi@debian:~$ ping 4.2.2.4
ping: connect: Network is unreachable

jadi@debian:~$ ip route add default via 192.168.70.1
RTNETLINK answers: Operation not permitted

jadi@debian:~$ sudo ip route add default via 192.168.70.1

jadi@debian:~$ ping 4.2.2.4
PING 4.2.2.4 (4.2.2.4) 56(84) bytes of data.
64 bytes from 4.2.2.4: icmp_seq=1 ttl=55 time=316 ms
64 bytes from 4.2.2.4: icmp_seq=2 ttl=55 time=434 ms
^C
--- 4.2.2.4 ping statistics ---
2 packets transmitted, 2 received, 0% packet loss, time 1002ms
rtt min/avg/max/mdev = 316.253/375.282/434.311/59.029 ms
```

We were able to ping the Internet after adding a default gateway using `ip route` command. It is important to know that we can define even more routes and control exactly where our packets should go based on their destinations.

tracepath & traceroute

This is a more advanced troubleshooting tool. It is like pinging all the servers between you and your destination one by one and see where our packets go wrong. Lets check how I reach google.com.

Copy

```
jadi@debian:~$ traceroute 4.2.2.4
traceroute to 4.2.2.4 (4.2.2.4), 30 hops max, 60 byte packets
 1  192.168.70.1 (192.168.70.1)  0.619 ms *  0.673 ms
```

```
 2  10.192.0.1 (10.192.0.1)  421.898 ms  728.657 ms  728.617 ms
 3  162.221.202.253 (162.221.202.253)  728.597 ms  728.564 ms  728.552 ms
 4  * * *
 5  207.35.48.241 (207.35.48.241)  728.289 ms  728.265 ms  728.251 ms
 6  agg2-vancouverbg_5-2-0.net.bell.ca (64.230.122.250)  728.344 ms agg1-vancouverbg_5-2-0.net.bell.ca (64.230.122.248)  612.097 ms  922.055 r
 7  * * *
 8  bx6-seattle_et-0/0/13_ae1.net.bell.ca (64.230.76.157)  921.662 ms  921.635 ms  921.566 ms
 9  ae96.edge6.Seattle1.Level3.net (4.16.2.13)  921.511 ms  921.372 ms  921.317 ms
10  * * ae6.4.edge2.SanJose1.level3.net (4.69.220.185)  920.786 ms
11  d.resolvers.level3.net (4.2.2.4)  601.295 ms  921.003 ms  920.970 ms
```

On the first step (1) I reach my own router. On the 2nd step, I'm at my ISPs local network and then will reach 162.blah.blah.blah. You can see in some cases the traceroute were able to do a **reverse DNS lookup** and find the hostname of the IPs and show them. After 11 hops, I reached my destination. The traceroute is a very useful tool in troubleshooting network and routing issues or checking the status of your network and paths.

> In some routers, the ping traffic is blocked, and you will see `* * *` at some steps because those servers are blocking ICMP traffic.

There is another command called `tracepath` which is very similar to `traceroute`. For the LPIC1 level, they are essentially the same!

ss & netstat

These commands can show a wide range of information about our network. The `ss` is the newer one and `netstat` is the older one. The two commands are replaceable in most cases and does the same things and even have similar option in most use cases. You can use the `netstat` to check your routing table:

Copy

```
jadi@debian:~$ netstat -nr
Kernel IP routing table
Destination     Gateway         Genmask         Flags   MSS Window  irtt Iface
0.0.0.0         192.168.70.1    0.0.0.0         UG        0 0          0 enp0s1
172.17.0.0      0.0.0.0         255.255.0.0     U         0 0          0 docker0
192.168.70.0    0.0.0.0         255.255.255.0   U         0 0          0 enp0s1
```

Or use the `ss` to see which port is in the LISTENING mode:

Copy

```
jadi@debian:~$ ss -na | grep LISTEN | grep tcp
tcp   LISTEN 0      128                                        127.0.0.1:631             0.0.0.0:*
tcp   LISTEN 0      100                                          0.0.0.0:25              0.0.0.0:*
tcp   LISTEN 0      128                                          0.0.0.0:22              0.0.0.0:*
tcp   LISTEN 0      100                                             [::]:25                 [::]:*
tcp   LISTEN 0      128                                             [::]:22                 [::]:*
tcp   LISTEN 0      128                                           [::1]:631                [::]:*
```

The `-na` switch will show all the open ports (including sockets) and here I'm checking only for the tcp ones in the LISTEN status.

> In these switches, `-n` stands for *numeric*, `-a` stands for *all ports* and `-r` stands for *routes*.

A super common combination is the `-tulpn` option set:

Copy

```
jadi@debian:~$ ss -tulpn
Netid             State              Recv-Q             Send-Q                        Local Address:Port                         Peer Addres
udp               UNCONN             0                  0                                   0.0.0.0:68                              0.0.0
udp               UNCONN             0                  0                                   0.0.0.0:631                             0.0.0
```

```
udp               UNCONN            0                 0                                0.0.0.0:58120                              0.0.0
udp               UNCONN            0                 0                                0.0.0.0:5353                               0.0.0
udp               UNCONN            0                 0                                   [::]:39822                                 [:
udp               UNCONN            0                 0                                   [::]:5353                                  [:
tcp               LISTEN            0                 128                            127.0.0.1:631                              0.0.0
tcp               LISTEN            0                 100                              0.0.0.0:25                               0.0.0
tcp               LISTEN            0                 128                              0.0.0.0:22                               0.0.0
tcp               LISTEN            0                 100                                 [::]:25                                 [:
tcp               LISTEN            0                 128                                 [::]:22                                 [:
tcp               LISTEN            0                 128                               [::1]:631                                 [:
```

netcat

The `nc` (or netcat) utility is used for just about anything under the sun involving TCP, UDP, or UNIX-domain sockets. It can open TCP connections, send UDP packets, listen on arbitrary TCP and UDP ports, do port scanning, and deal with both IPv4 and IPv6. Unlike `telnet`, `nc` can be used easily in the scripts and separates error messages onto standard error instead of sending them to standard output, as telnet does with some. This is a very capable command, and it is enough for you to be familiar with its general concept.

Here I'm creating a tcp listener on port 1337:

Copy

```
jadi@debian:~$ nc -l 1337
```

And here I'm opening a connection to that port and sending some data:

Copy

```
jadi@debian:~$ nc localhost 1337
Are you enjoying the LPIC?
```

And the message should be visible on the listening `nc`.

dig

The `dig` command is a DNS lookup tool. If you are having a problem with a domain name, you can check how it is being resolved to IPs; and by whom.

Copy

```
[jadi@debian ~]$ dig google.com

; <<>> DiG 9.9.5-11ubuntu1.3-Ubuntu <<>> google.com
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 50032
;; flags: qr rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 0

;; QUESTION SECTION:
;google.com.            IN    A

;; ANSWER SECTION:
google.com.        293    IN    A    216.58.214.46

;; Query time: 120 msec
;; SERVER: 4.2.2.4#53(4.2.2.4)
;; WHEN: Fri Apr 01 22:00:05 IRDT 2016
;; MSG SIZE  rcvd: 44
```

You can see that SERVER 4.2.2.4 is resolving google.com to 216.58.214.46.

109.4 Configure client side DNS

Weight: 2

Candidates should be able to configure DNS on a client host.

Key Knowledge Areas

- Query remote DNS servers.
- Configure local name resolution and use remote DNS servers.

- Modify the order in which name resolution is done.
- Debug errors related to name resolution
- Awareness of systemd-resolved

Terms and Utilities

- `/etc/hosts`
- `/etc/resolv.conf`
- `/etc/nsswitch.conf`
- `host`
- `dig`
- `getent`

DNS

We already know a lot about Domain Name System; A service which translates domain names (say yahoo.com) to IP addresses (say 206.190.36.45). A DNS server is used when you ping a server using its domain name. You have seen the config files for DNS and should know that the actual DNS server which is being used by the computer can be checked / changed (temporarily) from `/etc/resolv.conf`:

Copy

```
$ cat /etc/resolv.conf
# Generated by NetworkManager
nameserver 192.168.1.1
nameserver 4.2.2.4

$ ping x.org
PING x.org (131.252.210.176) 56(84) bytes of data.
64 bytes from annarchy.freedesktop.org (131.252.210.176): icmp_seq=1 ttl=45 time=338 ms
64 bytes from  annarchy.freedesktop.org (131.252.210.176): icmp_seq=2 ttl=45 time=333 ms
^C
--- x.org ping statistics ---
2 packets transmitted, 2 received, 0% packet loss, time 1000ms
rtt min/avg/max/mdev = 333.088/335.612/338.136/2.524 ms
```

`host`

This is a simple program to lookup DNS queries. This is a sample with no arguments:

Copy

```
$ host kernel.org
kernel.org has address 139.178.84.217
kernel.org has IPv6 address 2604:1380:4641:c500::1
kernel.org mail is handled by 10 smtp1.kernel.org.
kernel.org mail is handled by 10 smtp2.kernel.org.
kernel.org mail is handled by 10 smtp3.kernel.org.
```

As you can see, the `host` command returns all the records it can find for a specific domain. In this case it returns IPv4 (A), IPv6 (AAAA) and two Mail (MX) records.

> if you want to check only a specific record, you can provide it via `-t` swtich. So the `-t A` will only return back the IPv4 records.

dig

The `dig` tool is specifically build to query DNS servers. If you want to find out where x.org points to, you can do:

Copy

```
$ dig x.org

; <<>> DiG 9.10.3-P4-RedHat-9.10.3-12.P4.fc23 <<>> x.org
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 7483
;; flags: qr rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 512
;; QUESTION SECTION:
;x.org.                IN    A

;; ANSWER SECTION:
x.org.            1625    IN    A    131.252.210.176

;; Query time: 35 msec
;; SERVER: 192.168.1.1#53(192.168.1.1)
;; WHEN: Sun Apr 17 12:45:02 IRDT 2016
;; MSG SIZE  rcvd: 50
```

As you can see, `dig` did an **ip lookup** for `x.org` and told me that its IP is 131.252.210.176. The `1625` is called the **TTL** or *Time To Live* and show how many seconds this answer will be considered valid in cache. This command also tells us which server is used to query the answer (last 4 lines) and when and how long it took.

There is also a way to tell `dig` command what server it should use as the DNS vi `@<DNS-SERVER>`:

Copy

```
$ dig @8.8.8.8 google.com

; <<>> DiG 9.10.3-P4-RedHat-9.10.3-12.P4.fc23 <<>> @8.8.8.8 google.com
; (1 server found)
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 24313
;; flags: qr rd ra; QUERY: 1, ANSWER: 11, AUTHORITY: 0, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 512
;; QUESTION SECTION:
;google.com.            IN    A

;; ANSWER SECTION:
google.com.        112    IN    A    173.194.32.133
google.com.        112    IN    A    173.194.32.136
google.com.        112    IN    A    173.194.32.132
google.com.        112    IN    A    173.194.32.129
google.com.        112    IN    A    173.194.32.137
google.com.        112    IN    A    173.194.32.130
google.com.        112    IN    A    173.194.32.134
google.com.        112    IN    A    173.194.32.135
google.com.        112    IN    A    173.194.32.128
google.com.        112    IN    A    173.194.32.131
google.com.        112    IN    A    173.194.32.142

;; Query time: 238 msec
;; SERVER: 8.8.8.8#53(8.8.8.8)
;; WHEN: Sun Apr 17 13:03:38 IRDT 2016
;; MSG SIZE  rcvd: 215
```

Here I have asked dig to use `8.8.8.8` as its DNS and query `google.com`. You can see that I've got more than one answer (actually much more than one answer). My computer can randomly contact any of those IPs to reach the `google.com`. In other words, google.com is using more than one server/IP and `8.8.8.8` provides all of them when queried for that domain.

`/etc/hosts`

This file contains IP addresses and their correspondive names. This is kind of an static name resolution on your computer. Let's have a look:

Copy

```
$ head /etc/hosts
127.0.0.1    funlife localhost.localdomain    localhost clickadu.com
::1    funlife localhost6.localdomain6    localhost6
```

```
10.159.32.155 nsproxy
172.16.12.134 linuxclass wonderland
193.40.12.135 salma

87.106.233.90 gratis.vps

192.168.59.231 mass1
```

This file can be changed by root and will map some domain names (localhost, mass1, gratis.vps, ...) to some IP addresses. If I ping *mass1* on this computer.. lets see:

Copy

```
$ dig mass1

; <<>> DiG 9.10.3-P4-RedHat-9.10.3-12.P4.fc23 <<>> mass1
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NXDOMAIN, id: 39464
;; flags: qr rd ra; QUERY: 1, ANSWER: 0, AUTHORITY: 1, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 512
;; QUESTION SECTION:
;mass1.                IN    A

;; AUTHORITY SECTION:
.            600    IN    SOA    a.root-servers.net. nstld.verisign-grs.com. 2016041700 1800 900 604800 86400

;; Query time: 516 msec
;; SERVER: 192.168.1.1#53(192.168.1.1)
;; WHEN: Sun Apr 17 13:15:07 IRDT 2016
;; MSG SIZE  rcvd: 109

$ ping mass1
PING mass1 (192.168.59.231) 56(84) bytes of data.
From 85-15-16-103.shatel.ir (85.15.16.103) icmp_seq=1 Packet filtered
From 85-15-16-103.shatel.ir (85.15.16.103) icmp_seq=2 Packet filtered
```

My computer pings 192.168.59.231 when I go for mass1 even when the DNS can not find this name because that is defined in `/etc/hosts`.

nsswitch

The `/etc/nsswitch.conf` file tells the system about the priority of lookups, password checks, ... Lets have a look to make it clear:

Copy

```
$ cat /etc/nsswitch.conf
#
# /etc/nsswitch.conf
#
# An example Name Service Switch config file. This file should be
# sorted with the most-used services at the beginning.
#
# The entry '[NOTFOUND=return]' means that the search for an
# entry should stop if the search in the previous entry turned
# up nothing. Note that if the search failed due to some other reason
# (like no NIS server responding) then the search continues with the
# next entry.
#
# Valid entries include:
#
#    nisplus            Use NIS+ (NIS version 3)
#    nis            Use NIS (NIS version 2), also called YP
#    dns            Use DNS (Domain Name Service)
#    files            Use the local files
#    db            Use the local database (.db) files
#    compat            Use NIS on compat mode
#    hesiod            Use Hesiod for user lookups
#    [NOTFOUND=return]    Stop searching if not found so far
#

# To use db, put the "db" in front of "files" for entries you want to be
# looked up first in the databases
#
# Example:
#passwd:    db files nisplus nis
#shadow:    db files nisplus nis
#group:     db files nisplus nis
```

```
passwd:     files sss
shadow:     files sss
group:      files sss

#hosts:     db files nisplus nis dns
hosts:      files mdns4_minimal [NOTFOUND=return] dns myhostname mymachines

# Example - obey only what nisplus tells us...
#services:   nisplus [NOTFOUND=return] files
#networks:   nisplus [NOTFOUND=return] files
#protocols:  nisplus [NOTFOUND=return] files
#rpc:        nisplus [NOTFOUND=return] files
#ethers:     nisplus [NOTFOUND=return] files
#netmasks:   nisplus [NOTFOUND=return] files

bootparams: nisplus [NOTFOUND=return] files

ethers:     files
netmasks:   files
networks:   files
protocols:  files
rpc:        files
services:   files sss

netgroup:   files sss

publickey:  nisplus

automount:  files sss
aliases:    files nisplus
```

On the DNS line I have `hosts: files mdns4_minimal [NOTFOUND=return] dns myhostname mymachines`. This means when the system wants to find the IP address of a name, it first go for the `files` (/etc/hosts) and then for m**dns**4_minimal and then **dns** and so on. In this case if I add the `facebook.com` to my `/etc/hosts` like this:

Copy

```
127.0.0.1 facebook.com
```

And then point my browser to facebook.com, my computer will try to connect to the webserver on 127.0.0.1 instead of the real IP of Facebook.

getent

The `getent` command is a utility to get entries from Name Service Switch libraries (read /etc/nsswitch.conf). If you want to check what is the config of your hosts, you can do as follow.

Copy

```
$ getent hosts
127.0.0.1       funlife localhost.localdomain localhost clickadu.com
127.0.0.1       funlife localhost6.localdomain6 localhost6
10.159.32.155   nsnproxy
172.16.12.134   linuxclass wonderland
193.40.12.135   salma
87.106.233.90   gratisvps
192.168.59.231  mass1
192.168.59.232  mass2
192.168.59.233  mass3
192.168.59.234  mass4
192.168.59.235  mass5
192.168.59.236  mass6
192.168.59.237  mass7
192.168.59.238  mass8
192.168.59.239  mass9
127.0.0.1       frctlstartupfailure localtodoer localdeliv
127.0.0.1       frctlmeth
```

systemd-resolved

It should be noted that the `systemd` provides a DNS called `systemd-resolved`. It listens for DNS requests on `127.0.0.53` and answers back after consulgint the `/etc/systemd/resolv.conf` or `/etc/resolv.conf`.

Read more Here about `systemd-resolved`.

110.1 Perform security administration tasks

Weight: 3

Candidates should know how to review system configuration to ensure host security in accordance with local security policies.

Key Knowledge Areas

- Audit a system to find files with the suid/sgid bit set.
- Set or change user passwords and password aging information.
- Being able to use nmap and netstat to discover open ports on a system.
- Set up limits on user logins, processes and memory usage.
- Determine which users have logged in to the system or are currently logged in.
- Basic sudo configuration and usage.

Terms and Utilities

- `find`
- `passwd`
- `fuser`
- `lsof`
- `nmap`
- `chage`
- `netstat`
- `sudo`
- `/etc/sudoers`
- `su`
- `usermod`
- `ulimit`
- `who, w, last`

Users

sudo VS su

We've used `sudo` and `su` in practically all the chapters and now is the time to have a closer look! `su` switches you to some other account; a "substitute user identity". You get a new prompt with the new user account after successfully `su`ing to that account:

Copy

```
jadi@funlife ~$ whoami
jadi
jadi@funlife ~$ su -
Password:
root@funlife:~# whoami
root
root@funlife:~# su jadi -
jadi@funlife /root$ whoami
jadi
jadi@funlife /root$ exit
exit
root@funlife:~# whoami
root
root@funlife:~# exit
logout
jadi@funlife ~$ whoami
jadi
jadi@funlife ~$
```

Note that when running `su` you have to **provide the root password** to become root; or any other users password to become that user!

On the other hand, sudo asks for your own password and runs the command you gave it, with the root privileges. So `sudo ls` runs the ls command with the root privileges after asking for **your password**. Obviously you should have the *sudo right* to issue sudo. This is defined in `/etc/sudoers` file:

Copy

```
$ sudo cat /etc/sudoers
#
# This file MUST be edited with the 'visudo' command as root.
#
# Please consider adding local content in /etc/sudoers.d/ instead of
# directly modifying this file.
#
# See the man page for details on how to write a sudoers file.
#
Defaults    env_reset
Defaults    mail_badpass
Defaults    secure_path="/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin"

# Host alias specification

# User alias specification

# Cmnd alias specification

# User privilege specification
root    ALL=(ALL:ALL) ALL

# Members of the admin group may gain root privileges
%admin ALL=(ALL) ALL

# Allow members of group sudo to execute any command
%sudo    ALL=(ALL:ALL) ALL

# See sudoers(5) for more information on "#include" directives:

#includedir /etc/sudoers.d
```

Note the 2 important lines: - How root gets the right to run all the commands: `txt root ALL=(ALL:ALL) ALL` - And how the sudo and admin groups get rights to run commands as root: ```txt # Members of the admin group may gain root privileges %admin ALL=(ALL) ALL

Allow members of group sudo to execute any command %sudo ALL=(ALL:ALL) ALL ```

The `ALL:ALL` means these users can run as any user and any group. The last ALL tells the sudo that these users/groups can run ALL commands. It is possible to put `/bin/ping` in the last part to tell sudo that this user can run only ping as root, Like below:

```
txt username ALL=(ALL) /bin/ping
```

> The /etc/sudoers file is very important and breaking it will make major problems. to prevent you from adding un-interpretable lines in that file, the `visudo` command should be used instead of `vi /etc/sudoers`. This tool will check your edits to make sure that sudo

command can understand them.

Now we know what `sudo su -` means. The sudo tells the system to run the `su -` command with the root access. It asks your password and runs the `su -` as the root if you have sudo access. The `su` commands changes your user to root and `-` switch load the root environment variables. This way you can become root using your own password via running `su` with `sudo`.

checking the users in the system

If you need to check who is in your system (and to some extend what they are doing) you can use these commands:

Copy

```
$  w
 22:03:37 up 3 days,  5:33, 13 users,  load average: 1.48, 1.12, 1.19
USER     TTY      FROM             LOGIN@   IDLE   JCPU   PCPU WHAT
jadi     tty7     :0               Wed16    3days  2:30m  1.96s /sbin/upstart --user
jadi     pts/18   :0               Wed16    3:04m  1:02   1:02  /usr/bin/python manage.py runserver 0.0.0.0:8000
jadi     pts/19   :0               Wed16    1:11m  0.35s  0.35s /bin/bash
root     tty2                      Wed16    3days  0.07s  0.03s -bash
jadi     pts/21   :0               08:41   45:37   0.06s  0.06s /bin/bash
jadi     pts/23   :0               Thu11   46:49   0.25s  0.23s ssh startups
jadi     pts/21   funlife          Fri22   45:37   0.06s  0.06s /bin/bash
jadi     pts/25   :0               10:17   31:37   0.07s  5.81s /usr/bin/python /usr/bin/x-terminal-emulator
jadi     pts/26   :0               21:39    0.00s  0.07s  0.00s w
jadi     pts/27   :0               21:55    8:09   0.01s  0.01s /bin/bash
```

You have a line for each logged in users (every single shell window is a separated login).

Another useful command is `who`. Lets check it:

Copy

```
$ who
jadi     tty7         2016-06-01 16:30 (:0)
jadi     pts/17       2016-06-01 16:30 (funlife)
jadi     pts/2        2016-06-01 16:32 (:0)
jadi     pts/18       2016-06-01 16:32 (:0)
jadi     pts/19       2016-06-01 16:33 (:0)
root     tty2         2016-06-01 16:36
jadi     pts/21       2016-06-04 08:41 (:0)
jadi     pts/22       2016-06-01 18:37 (funlife)
jadi     pts/23       2016-06-02 11:41 (:0)
jadi     pts/21       2016-06-03 22:22 (funlife)
jadi     pts/25       2016-06-04 10:17 (:0)
jadi     pts/26       2016-06-04 21:39 (:0)
jadi     pts/27       2016-06-04 21:55 (:0)
```

As you can see both these commands tell you when was the time that the user logged into the system but does not show the logged out people (because they are not on the system anymore!). If you need that data use the `last` command:

Copy

```
~$ last | head
jadi     pts/27       :0               Sat Jun  4 21:55    gone - no logout
jadi     pts/26       :0               Sat Jun  4 21:39    gone - no logout
jadi     pts/26       :0               Sat Jun  4 18:55 - 19:42  (00:46)
jadi     pts/25       :0               Sat Jun  4 10:17    gone - no logout
jadi     pts/26       :0               Sat Jun  4 09:25 - 09:26  (00:00)
jadi     pts/26       :0               Sat Jun  4 09:25 - 09:25  (00:00)
jadi     pts/25       :0               Sat Jun  4 08:52 - 09:27  (00:35)
jadi     pts/21       :0               Sat Jun  4 08:41    gone - no logout
jadi     pts/21       funlife          Fri Jun  3 22:22 - 08:41  (10:18)
jadi     pts/21       :0               Fri Jun  3 18:44 - 19:47  (01:03)
```

there is a way to check the failed logins too: `last -f /var/log/btmp`

Password Management

The `passwd` command is used to update / change the password of the users.

Copy

```
➜  passwd
Changing password for jadi.
Current password:
New password:
Retype new password:
passwd: password updated successfully
➜  sudo passwd jadi
```

```
New password:
BAD PASSWORD: The password is shorter than 8 characters
Retype new password:
passwd: password updated successfully
```

You can also use this command to check the status of a user:

Copy

```
➜  ~ passwd -S
jadi P 2023-09-14 0 99999 7 -1
```

It says my user is jadi, I have a valid *P*assword (could have been *L*ocked or *NP* for no password), my last password change time, minimum and maximum age of my password, warning period before password expiry & allowed password inactivity in days. The `passwd` command can also be used to `--lock` (or `-l`) a user, `--expire` (or `-e`) a user or `--unlock` (or `-u`) a user.

> As you've already see in the user management section, to change the user shell, home, ... you should use the `usermod` command

But to properly check/change the password age and configurations of users, the `chage` utility should be used. Run it with `-l` for `list`:

Copy

```
➜  ~ chage -l jadi
Last password change                    : Sep 14, 2023
Password expires                        : never
Password inactive                       : never
Account expires                         : never
Minimum number of days between password change      : 0
Maximum number of days between password change      : 99999
Number of days of warning before password expires   : 7
```

or without any switches for interactive mode:

Copy

```
➜  ~ chage jadi
chage: Permission denied.
➜  ~ sudo chage jadi
Changing the aging information for jadi
Enter the new value, or press ENTER for the default

    Minimum Password Age [0]:
    Maximum Password Age [99999]:
    Last Password Change (YYYY-MM-DD) [2023-09-14]:
    Password Expiration Warning [7]: 3
    Password Inactive [-1]:
    Account Expiration Date (YYYY-MM-DD) [-1]:
```

You can also use switches to directly change values. Say `-m` for --mindays or `-M` for --maxdays.

suid and guid

We've already covered `suid`; In short, when the suid bit is set on an executable file, the user will run with the access level of the owner of the file (and not the runner). Have a look at the `ping` command:

Copy

```
➜  ~ type passwd
passwd is /usr/bin/passwd
➜  ~ ls -ltrh /usr/bin/passwd
-rwsr-xr-x 1 root root 63K Nov 23  2022 /usr/bin/passwd
```

The `s` on the access rights part, will run the file using the owners (here its root) access; regardless of whom is running it. This is needed because the `passwd` commands needs to be able to change the passwords in the *shadow* file even if a normal user runs it. But what happens if someone changes the suid of the vi command? let's see who owns the `vi`:

Copy

```
jadi@funlife ~$ type vi
vi is /usr/bin/vi
jadi@funlife ~$ ls -ltrh /usr/bin/vi
lrwxrwxrwx 1 root root 20 Jun  1 12:52 /usr/bin/vi -> /etc/alternatives/vi
```

The `vi` is owned by root, so if the suid bit is set, vi will always be run as root! In that case, anybody will be able to edit any file! So if you are a hacker and get a temporary root, its enough to copy vi somewhere (with a different unsuspicious name) and give it the `suid` access. Now you will be able to run it with normal users and modify system files when needed! That's why a system admin should be able to check her systems executable files with `suid` bit if needed:

Copy

```
$sudo find / -perm -u+s
```

Here, the `-perm -u+s` tells `find` to search for file which has `suid` on `user`. For more info, check the `man find` and search for `perm`.

> same applies for guid. if the guid is set, the file will be run with access of its group

user limits

The resources on a Linux machine can be managed for users by the `ulimit` command. It is part of the PAM system. If you want to check the limits on the system run:

Copy

```
~$ ulimit -a
core file size          (blocks, -c) 0
data seg size           (kbytes, -d) unlimited
scheduling priority             (-e) 0
file size               (blocks, -f) unlimited
pending signals                 (-i) 47457
max locked memory       (kbytes, -l) 64
max memory size         (kbytes, -m) unlimited
open files                      (-n) 1024
pipe size            (512 bytes, -p) 8
POSIX message queues     (bytes, -q) 819200
real-time priority              (-r) 0
stack size              (kbytes, -s) 8192
cpu time               (seconds, -t) unlimited
max user processes              (-u) 47457
virtual memory          (kbytes, -v) unlimited
file locks                      (-x) unlimited
```

and you can change them like this:

Copy

```
$ ulimit -t 1
```

This will limit the CPU TIME of any process to 1 second. If you use more than that, the process will be killed automatically (by PAM module). Please note that your walls clock time is different than CPU time. To see how much CPU time a process uses run it like this:

Copy

```
$ time firefox
```

Changing the ulimit as we did is a temporary thing. It only persists in that specific shell.

To change the ulimits system-wide:

Copy

```
$ cat /etc/security/limits.conf
# /etc/security/limits.conf
#
#Each line describes a limit for a user in the form:
#
#<domain>        <type>  <item>  <value>
```

```
#
#Where:
#<domain> can be:
#        - a user name
#        - a group name, with @group syntax
#        - the wildcard *, for default entry
#        - the wildcard %, can be also used with %group syntax,
#                for maxlogin limit
#        - NOTE: group and wildcard limits are not applied to root.
#          To apply a limit to the root user, <domain> must be
#          the literal username root.
#
#<type> can have the two values:
#        - "soft" for enforcing the soft limits
#        - "hard" for enforcing hard limits
#
#<item> can be one of the following:
#        - core - limits the core file size (KB)
#        - data - max data size (KB)
#        - fsize - maximum filesize (KB)
#        - memlock - max locked-in-memory address space (KB)
#        - nofile - max number of open files
#        - rss - max resident set size (KB)
#        - stack - max stack size (KB)
#        - cpu - max CPU time (MIN)
#        - nproc - max number of processes
#        - as - address space limit (KB)
#        - maxlogins - max number of logins for this user
#        - maxsyslogins - max number of logins on the system
#        - priority - the priority to run user process with
#        - locks - max number of file locks the user can hold
#        - sigpending - max number of pending signals
#        - msgqueue - max memory used by POSIX message queues (bytes)
#        - nice - max nice priority allowed to raise to values: [-20, 19]
#        - rtprio - max realtime priority
#        - chroot - change root to directory (Debian-specific)
#
#<domain>      <type>  <item>         <value>
#

#*               soft    core            0
#root            hard    core            100000
#*               hard    rss             10000
#@student        hard    nproc           20
#@faculty        soft    nproc           20
#@faculty        hard    nproc           50
#ftp             hard    nproc           0
#ftp             -       chroot          /ftp
#@student        -       maxlogins       4

# End of file
```

soft limits can be changed by the user but hard limits are the real stop points.

Open Ports

netstat, fuser and lsof

On module 109.1 we talked about ports. Ports are like wholes in our systems used by programs to listen to the outside world. If I'm running a web server on my computer I should have a port open so people can ask that server "please show me your index.html". Many malwares open ports to let the attacker to communicate with them. It is important to check your computer for open ports time to time. The older command for this is the `netstat`; using the `-na` or `-ap` or `-tuna` switch.. I'm sure **tuna** is easy to remember if you have every enjoyed a tuna sandwich.

Copy

```
jadi@funlife ~$ netstat -tuna
Active Internet connections (servers and established)
Proto Recv-Q Send-Q Local Address           Foreign Address         State
tcp        0      0 127.0.0.1:3306          0.0.0.0:*               LISTEN
tcp        0      0 0.0.0.0:80              0.0.0.0:*               LISTEN
tcp        0      0 127.0.1.1:53            0.0.0.0:*               LISTEN
tcp        0      0 127.0.0.1:9050          0.0.0.0:*               LISTEN
tcp       25      0 192.168.59.9:49934      192.168.59.192:139      CLOSE_WAIT
tcp        0      0 127.0.0.1:60228         127.0.0.1:1080          ESTABLISHED
tcp        0      0 192.168.1.35:55324      159.203.148.169:8385    ESTABLISHED
tcp        0      0 127.0.0.1:59590         127.0.0.1:1080          ESTABLISHED
tcp        0      0 127.0.0.1:60212         127.0.0.1:1080          ESTABLISHED
tcp        0      0 192.168.1.35:54220      159.203.148.169:8385    ESTABLISHED
tcp        0      0 127.0.0.1:57186         127.0.0.1:1080          ESTABLISHED
tcp        0      0 192.168.1.35:49574      173.194.122.231:443     ESTABLISHED
tcp        0      0 127.0.0.1:59002         127.0.0.1:1080          ESTABLISHED
udp        0      0 0.0.0.0:54502           0.0.0.0:*
udp        0      0 0.0.0.0:5353            0.0.0.0:*
udp        0      0 0.0.0.0:5353            0.0.0.0:*
```

All the `LISTEN` ports are servers; they are LISTENING for new incoming connections. The `ESTABLISHED` connections are the active connections between your computer and another computer. In these tables `0.0.0.0` dictates *any address* or *any interface*.

Other useful tools are `ss` for the same purpose as you saw on chapter 109 and `lsof` and `fuser`. The `lsof` is already discussed in previous sections. It shows the open files on the system and having in mind that *everything in Linux is a file or a process* you can conclude that this command should be able to display open connections too; and you are right:

Copy

```
# lsof -i
COMMAND     PID       USER   FD   TYPE  DEVICE SIZE/OFF NODE NAME
privoxy     806    privoxy    4u  IPv4   16130      0t0  TCP funlife:8118 (LISTEN)
cups-brow   903       root    8u  IPv4   17477      0t0  UDP *:ipp
mysqld      971      mysql   19u  IPv4   20875      0t0  TCP funlife:mysql (LISTEN)
tor        1038 debian-tor    6u  IPv4   19155      0t0  TCP funlife:9050 (LISTEN)
dnsmasq    1260     nobody   11u  IPv4 1910037      0t0  UDP *:18666
adb        1278       jadi    5u  IPv4  579541      0t0  TCP funlife:5037 (LISTEN)
chromium-  2891       jadi   88u  IPv4  813611      0t0  TCP 192.168.1.35:45702->do-13.lastpass.com:https (ESTABLISHED)
chromium-  2891       jadi  126u  IPv4 1907389      0t0  TCP 192.168.1.35:50642->ntt-2.lastpass.com:https (ESTABLISHED)
chromium-  2891       jadi  133u  IPv4 1909733      0t0  TCP 192.168.1.35:50644->ntt-2.lastpass.com:https (ESTABLISHED)
chromium-  2891       jadi  268u  IPv4  785289      0t0  TCP 192.168.1.35:60736->lf-in-f188.1e100.net:5228 (ESTABLISHED)
python     4925       jadi    4u  IPv4  658287      0t0  TCP funlife:8000 (LISTEN)
Telegram   4943       jadi   39u  IPv4  773463      0t0  TCP 192.168.1.35:44732->149.154.175.50:https (ESTABLISHED)
dhclient   9984       root    6u  IPv4  787885      0t0  UDP *:bootpc
nginx     11095       root    6u  IPv4   17998      0t0  TCP *:http (LISTEN)
nginx     11099   www-data    7u  IPv6   17999      0t0  TCP *:http (LISTEN)
chrome    14264       jadi  114u  IPv4  788089      0t0  UDP *:mdns
chrome    14264       jadi  126u  IPv4 1872872      0t0  TCP funlife:60370->funlife:socks (ESTABLISHED)
chrome    14264       jadi  138u  IPv4 1908382      0t0  TCP funlife:60408->funlife:socks (ESTABLISHED)
```

Wow! This command shows the command, PID, user running it and source and destination IP and tells of if this is a LISTENING or STABLISHED connection.

If you want to check which process is using port 80, you can grep the output of any above commands or simply use the `fuser` (file user; who uses this file) command to find all the PIDs related to that specific port. A common switch is `-v` which goes verbose.

Copy

```
➜  bin sudo fuser 22/tcp -v
                     USER        PID ACCESS COMMAND
22/tcp:              root          1 F.... systemd
```

nmap

`nmap` is the one of the hackers beloved tools! You can nmap a server to find out about a lot of data about that server:

Copy

```
# nmap localhost

Starting Nmap 7.01 ( https://nmap.org ) at 2016-06-04 21:32 IRDT
Nmap scan report for localhost (127.0.0.1)
```

```
Host is up (0.0000070s latency).
rDNS record for 127.0.0.1: funlife
Not shown: 995 closed ports
PORT     STATE SERVICE
80/tcp   open  http
1080/tcp open  socks
3306/tcp open  mysql
8000/tcp open  http-alt
9050/tcp open  tor-socks

Nmap done: 1 IP address (1 host up) scanned in 1.66 seconds
root@funlife:~#
```

In the most basic form, nmap checks all the open ports from 1 to 1000 and prints the results. There are a lot of switches to find other information about the hosts and they are used by every single hacker who wants to examine a servers status.

Do a quick search for nmap oneliners and you will find dozens of cool commands and tricks for it. Its so cool that its website has a section about nmap in movies!

110.2 Setup host security

Weight: 3

Candidates should know how to set up a basic level of host security.

Key Knowledge Areas

- Awareness of shadow passwords and how they work.
- Turn off network services not in use.
- Understand the role of TCP wrappers.

Terms and Utilities

- `/etc/nologin`
- `/etc/passwd`
- `/etc/shadow`
- `/etc/xinetd.d/`
- `/etc/xinetd.conf`
- `systemd.socket`
- `/etc/inittab`
- `/etc/init.d/`
- `/etc/hosts.allow`
- `/etc/hosts.deny`

shadow passwords

The `/etc/passwd` is already discussed. It contains the passwords of the users but there is a logical problem: if a user should be able to change her own password, he should have access to this file and if this is the case, he can see other people's passwords. This is not interesting even when the passwords are hashed (shown as a more complex form using a one way function).

Copy

```
$ ls -ltrh /etc/passwd
-rw-r--r-- 1 root root 2.5K Jun  5 19:14 /etc/passwd
```

To prevent this the `/etc/shadow` file is introduced. In modern systems, we only show a `*` at the location of the password in `/etc/passwd` and store the real password in `/etc/shadow` which has a very limited file access:

Copy

```
jadi@funlife ~$ grep jadi /etc/passwd
jadi:x:1000:1000:jadi,,,:/home/jadi:/bin/bash
jadi@funlife ~$ grep jadi /etc/shadow
grep: /etc/shadow: Permission denied
jadi@funlife ~$ sudo grep jadi /etc/shadow
[sudo] password for jadi:
jadi:$6$bp01DBX.$I6dt4pz8GeXJl6asgPeKhSdepf40bgepTz8zwB3HFmN56SdcsxjTETdZAmRt17biwMYOI7SoGFOXssHqeNFgw/:16963:0:99999:7:::
jadi@funlife ~$ sudo ls -ltrh /etc/shadow
-rw-r----- 1 root shadow 1.5K Jun 11 17:36 /etc/shadow
```

`/etc/nologin`

The `/etc/nologin` is a cool file! If you create and write something in it, the content will be shown to any person who tries to login into the system and with that file the login attempt will fail. It is useful for maintenance time. Delete this file and the users will be able to login again.

> the root user will be able to login even in the presence of /etc/nologin

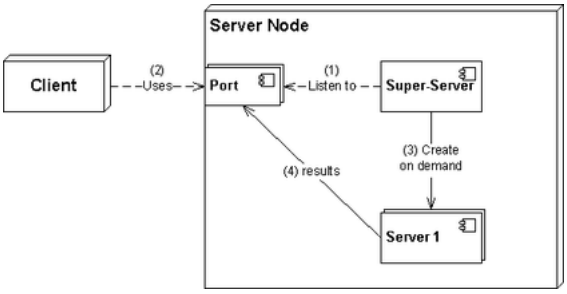
Please also note that there is a dummy shell called `nologin` and you can set it as shell for any user you want to prevent from being able to login into the system via a shell. Note that such a user still has an active account and will be able to use other services (say email or ftp) but wont be able to enter the shell.

Copy

```
sudo usermod -s /sbin/nologin baduser
```

super-servers

A super-server or sometimes called a service dispatcher is a type of daemon running mostly on Unix-like systems for security and resource management reasons. It listens for requests its configured for and starts the target services when needed to answer the requests. This adds a layer of security to your communications and also lets some of the services to be inactive when we do not need them. You can see a super server or superdaemon as a TCP (and UDP or even ICMP) wrapper around other services.



Few GNU/Linuxes use TCP wrappers like `xinetd` these days but you might see it in some installations or traces of it in your `/etc/xinet.d`. If needed it is also possible to configure the `systemd.socket` as a TCP wrapper for other services.

Here is a sample `xinetd` configuration file:

Copy

```
service telnet
{
        disable         = no
        flags           = REUSE
        socket_type     = stream
        wait            = no
        user            = root
        server          = /usr/sbin/in.telnetd
        log_on_failure  += USERID
        no_access       = 10.0.1.0/24
        log_on_success  += PID HOST EXIT
        access_times    = 09:45-16:15
}
```

If we change the `disable` to `yes` and restart the xinetd, the telnet daemon will start running. There are a few files to control xinetd related files.

As mentioned, `xinetd` is replaced by the `systemd.socket` units. Some services like `ssh` and `cups` might have a socket unit alongside the service unit in your distribution. In this case its enough to stop & disable the `ssh.service` and start the `ssh.docekt` instead. Now the `systemd.socekt` is acting as a wrapper around the port 22 and IF someones needs the service, starts the ssh server to answer.

`/etc/hosts.allow` & `/etc/hosts.deny`

These two files will allow or deny access from specific hosts. Its logic is like cron.deny and cron.allow. If something is allowed, everything else is denied but if you add something to the `/etc/hosts.deny`, only that specific thing is denied (and every other thing is allowed).

Copy

```
jadi@funlife ~$ cat /etc/hosts.allow
# /etc/hosts.allow: list of hosts that are allowed to access the system.
#                   See the manual pages hosts_access(5) and hosts_options(5).
#
# Example:    ALL: LOCAL @some_netgroup
#             ALL: .foobar.edu EXCEPT terminalserver.foobar.edu
#
# If you're going to protect the portmapper use the name "rpcbind" for the
# daemon name. See rpcbind(8) and rpc.mountd(8) for further information.
#

vsftpd: 10.10.100.
```

How the `vsftpd` knows about this file? Because it uses the `libwrap` library in its source and the libwrap understands the wrapper tools. You can check this claim by searching for `libwrap` in the list of libraries `vsftpd` uses:

Copy

```
➜  ~ ldd /usr/sbin/vsftpd | grep libwrap
    libwrap.so.0 => /lib/x86_64-linux-gnu/libwrap.so.0 (0x00007fb293921000)
```

Here the `vsftpd` service is only allowed from 10.10.100.* . It is possible to use `ALL` as the service name to allow or deny ALL services.

> after changing this file, xinetd should be restarted

As mentioned, super servers are not being used anymore and most distributions use standalone services.

Removing unused services

Based on your distribution, you can check the running services using the `service` or `systemctl` command. If you are using the SysV init system (older distros mainly) or you have the compatibility tools installed, do as follow:

Copy

```
~ sudo service --status-all
 [ - ]  alsa-utils
 ....
 [ + ]  ufw
 [ + ]  unattended-upgrades
 [ + ]  uuidd
 [ + ]  vpn-unlimited-daemon
 [ + ]  vsftpd
 [ - ]  whoopsie
 [ - ]  x11-common
```

On a RedHat based machine, you can stop a service via:

Copy

```
sudo chkconfig vsfptd off
```

and On debian machines:

Copy

```
sudo update-rc.d vsftpd remove
```

If you are using the systemd, do as follow to check and stop / disable a service:

Copy

```
systemctl list-units --state active --type service
systemctl status
systemctl disable vsftpd.service --now
```

Please also remember that on older systems, we used to have all the Init scripts in `/etc/init.d` and `/etc/rcX.d` folders. There were also a `/etc/inittab` file which was a configuration file for initializing a Linux system using SysV . It contains lines in this format:

This would tell the init system to do `actions` on the `process` on a specific `runlevel`. For example:

Copy

```
id:runlevel:action:process
```

It tells the init to start (and respawn if killed), the `mingetty tty1` command on runlevels `2`, `3`, `4`, & `5`.

Copy

```
1:2345:respawn:/sbin/mingetty tty1
```

As the final note, this was the most important line in the `inittab` file because it tells the init to start in run level 3.

Copy

```
id:3:initdefault:
```

For more information about `runlevels` , please refer to Chapter 101.3.

110.3 Securing data with encryption

Weight: 4

The candidate should be able to use public key techniques to secure data and communication.

Key Knowledge Areas

- Perform basic OpenSSH 2 client configuration and usage.
- Understand the role of OpenSSH 2 server host keys.
- Perform basic GnuPG configuration, usage and revocation.
- Use GPG to encrypt, decrypt, sign and verify files.
- Understand SSH port tunnels (including X11 tunnels).

Terms and Utilities

- ssh
- ssh-keygen
- ssh-agent
- ssh-add
- ~/.ssh/id_rsa and id_rsa.pub
- ~/.ssh/id_dsa and id_dsa.pub
- ~/.ssh/id_ecdsa and id_ecdsa.pub
- ~/.ssh/id_ed25519 and id_ed25519.pub
- /etc/ssh/ssh_host_rsa_key and ssh_host_rsa_key.pub
- /etc/ssh/ssh_host_dsa_key and ssh_host_dsa_key.pub
- /etc/ssh/ssh_host_ecdsa_key and ssh_host_ecdsa_key.pub
- /etc/ssh/ssh_host_ed25519_key and ssh_host_ed25519_key.pub
- ~/.ssh/authorized_keys
- ssh_known_hosts
- gpg
- gpg-agent
- ~/.gnupg/

Key Pairs

In traditional cryptography, the symmetric keys were used: both parties had a shared password. The data were encrypted with that password and then decrypted using the same password. But in 1976, a new idea came into the view: what if we create 2 keys (lets call them the Private Key and the Public Key) in a way that only people who has the Public Key, be able to open whatever which is encrypted with the Private key? The scientists made this a reality and nowawayds, most of our encryptions are being done via these **Key Pair**s. When generating a key pair, we generate two keys using a computer algorithm in the way that any message which is encrypted using one, can be opened only using the other. These are called Public & Private key. You publish the public key to your friends or even publicly on the net and if someone wants to send you an encrypted message, she encrypts it using your public key and send it to you with any means (or published the encoded message on the intneret) and only and only you will be able to open it, because you are the one who has the Private key!

> great point about Public / Private key is that the data can be transmitted over the internet with no fear of hackers or governments. You are publishing your key to the world, some one picks it and uses it to encrypt some data and sent the result to you. People can see that you are receiving "some data" but they can not encrypt it because they do not have the private key needed to decrypt it.

ssh keys

The same technology (Public key cryptography or asymetric cryptography) can be used in most of the network communications too. In fact the very `ssh` works based on this concept. It is used to authenticate hosts and secure the traffic.

Copy

```
➜  ~ ssh 192.168.70.2
The authenticity of host '192.168.70.2 (192.168.70.2)' can't be established.
ED25519 key fingerprint is SHA256:4Wp2zz6sgPAhnbqhkNjOd6QDpNQ4jvjX7qAzslPX09U.
This key is not known by any other names
Are you sure you want to continue connecting (yes/no/[fingerprint])? y
Please type 'yes', 'no' or the fingerprint: yes
Warning: Permanently added '192.168.70.2' (ED25519) to the list of known hosts.
jadi@192.168.70.2's password:
Linux debian 6.4.0-2-arm64 #1 SMP Debian 6.4.4-3 (2023-08-08) aarch64

The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
You have no mail.
Last login: Sun Sep 17 04:43:18 2023 from 192.168.70.1
jadi@debian:~$
```

Above the server is showing us the fingerprint of its ED25519 key and asking us to approve it. From now on, our system wont warn us for the same key with the same server. BUT if the fingerprint (so the key) of the same server is changed, that will be considered a serious case:

Copy

```
➜  ~ ssh 192.168.70.2
Host key verification failed.
@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
@    WARNING: REMOTE HOST IDENTIFICATION HAS CHANGED!     @
@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
IT IS POSSIBLE THAT SOMEONE IS DOING SOMETHING NASTY!
Someone could be eavesdropping on you right now (man-in-the-middle attack)!
It is also possible that a host key has just been changed.
The fingerprint for the ED25519 key sent by the remote host is
SHA256:4Wp2zz6sgPAhnbqhkNjOd6QDpNQ4jvjX7qAzslPX09U.
Please contact your system administrator.
Add correct host key in /Users/jadi/.ssh/known_hosts to get rid of this message.
Offending RSA key in /Users/jadi/.ssh/known_hosts:548
Host key for 192.168.70.2 has changed and you have requested strict checking.
```

If you want to solve this - after making sure that this is not an attack - you have to remove the data of 192.168.70.2 from the `.ssh/known_hosts` file. Fortunately you do not need to directly edit the file. Its enough to use the `ssh-keygen` command:

Copy

```
➜  ~ ssh-keygen -R 192.168.70.2
# Host 192.168.70.2 found: line 547
# Host 192.168.70.2 found: line 548
# Host 192.168.70.2 found: line 549
/Users/jadi/.ssh/known_hosts updated.
Original contents retained as /Users/jadi/.ssh/known_hosts.old
```

creating your own key pairs

You can easily create as many key pairs want; Even with different algorithms (defined by `-t` switch and including dsa, ecdsa, ecdsa-sk, ed25519, ed25519-sk, rsa). Common choices are *rsa* which is the default and *ecdsa* which is used by bitcoin!

Lets create an ecdsa key pair on our machine. We can use this to login into the servers without providing a password or sign/encrypt messages.

Copy

```
jadi@debian:~$ ssh-keygen -t ecdsa
Generating public/private ecdsa key pair.
Enter file in which to save the key (/home/jadi/.ssh/id_ecdsa):
/home/jadi/.ssh/id_ecdsa already exists.
Overwrite (y/n)? y
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/jadi/.ssh/id_ecdsa
Your public key has been saved in /home/jadi/.ssh/id_ecdsa.pub
The key fingerprint is:
SHA256:WvJu5D7ns0a2z0sB6l8ZrCXnAgxSmb2kSwgo/lK0QYA jadi@debian
The key's randomart image is:
+---[ECDSA 256]---+
|.oo.  .+         |
|E .o .o o        |
|o ..+..o ..      |
| . o..oo.. o     |
|  o  ...S . *    |
| . .  .*..o* +   |
```

```
|  .   .oooo.=    |
|       .+.==     |
|       oo=+++.   |
+----[SHA256]-----+
```

and since its free, lets create an rsa key too:

Copy

```
jadi@debian:~$ ssh-keygen
Generating public/private rsa key pair.
Enter file in which to save the key (/home/jadi/.ssh/id_rsa):
/home/jadi/.ssh/id_rsa already exists.
Overwrite (y/n)? y
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/jadi/.ssh/id_rsa
Your public key has been saved in /home/jadi/.ssh/id_rsa.pub
The key fingerprint is:
SHA256:Q67Ob1DOfmF3gG449OfNLHXxE5QEH4B/UzHR8ycAHao jadi@debian
The key's randomart image is:
+---[RSA 3072]----+
|          .oo++*=|
|           oo .+=|
|        . ......+|
|       oo.. ...=o|
|       =E+   ...*|
|      ..=.* o o.o|
|      .o + = * ..|
|     o  o . o +  |
|      oo..   .   |
+----[SHA256]-----+
```

That fun *Image* is what you can show your friends so they can make sure that they are using the correct key.

> Above, I did not set a password for the keys. If I did, I had to provide that password everytime I wanted to use that key

These keys are saved in users `~/.ssh` directory. As you can guess, the system wide keys used for ssh server are located at `/etc/ssh`.

Key based / Password less login

The ssh server can be configured to check your identity using your keys. Its enough to save your public-key on the server's user account and tell the server to check for key-based logins too. In this case, your ssh client will provide your private key to the ssh server as a proof of identity and you will be able to enter the user without providing passwords.

First copy your *public* key and login into the server. Open the `~/.ssh/authorized_keys` file and add yours. Also make sure that the `/etc/ssh/sshd_config` contains the `PubkeyAuthentication yes`. Now log out and on the next login you should be able to login without password! This is super userful when you are managing many servers or want to automate `scp` copies and other stuff. Its also more secure from the sysadmin point of view because if I need to let a new user to login into my system, I do not need to share the password or communicate the password on any channel. Its enough to ask the person for his **public key** and add it to the users `.ssh/authorized_keys` file.

Oh! and you do not need to do the *key copying* manually; just use the `ssh-copy-id` command:

Copy

```
➜  ~ ssh-copy-id 192.168.70.2
/usr/bin/ssh-copy-id: INFO: Source of key(s) to be installed: "/Users/jadi/.ssh/id_dsa.pub"
/usr/bin/ssh-copy-id: INFO: attempting to log in with the new key(s), to filter out any that are already installed
/usr/bin/ssh-copy-id: INFO: 1 key(s) remain to be installed -- if you are prompted now it is to install the new keys
jadi@192.168.70.2's password:

Number of key(s) added:        1

Now try logging into the machine, with:   "ssh '192.168.70.2'"
and check to make sure that only the key(s) you wanted were added.
```

ssh-agent

The `ssh-agent` works like a key manager for your ssh keys. It keeps your private keys in memory and provides them when needed. For example if you have set a password on your key, you have to type the password every single time you are going to use that key. To make this easier using ssh-agent, you have to rush a shell with ssh-agent and add all your key to the agent:

Copy

```
jadi@debian:~$ ssh-agent /bin/bash
jadi@debian:~$ ssh-add
Identity added: /home/jadi/.ssh/id_rsa (jadi@debian)
```

```
Identity added: /home/jadi/.ssh/id_ecdsa (jadi@debian)
Identity added: /home/jadi/.ssh/id_dsa (/home/jadi/.ssh/id_dsa)
```

Now, the agent knows about your keys and wont ask for the passwords anymore.

`ssh-agent` can also *transfer* your keys safely in memory to other servers. Say you need to ssh to server A and while on server A, use your key to ssh to server B (or do a git pull using your private keys). The keys are on server A and its not safe to actually and physically copy them to server B. In this case you cal run `ssh-agent` and have your keys even when you are on server B.

ssh tunnels

The ssh can also be used for *tunnelling*. Its a very fun concept and a super userful tool in the hands of a network Ninja! Honestly we use it all the time to solve complicated problems.

As the name sggests, ssh tunelling *tunnels* the data between machines. Look at this example:

Copy

```
ssh -L 9000:hckrnews.com:80 root@5.161.197.79
```

Here I'm telling my computer to ssh using user `root` (which is not a great idea) to the `5.161.197.79` server; AND create a **local tunnel** (`-L`) from on my machine toward the `hckrnews.com` port `80` through that machine. Now if I connect to `localhost:9000` **on my machine**, the request will be tunnels through `5.161.197.79` toward hckrnews.com port 80. You can try it with `curl localhost:9000`.

Why this is useful? Say you have a program on your server which only answers back to the local requests (and not the internet). Using local forwarding you can forward a port on your computer to port which programs works on and use the program on your own machine!

There is also the concept of **Remote Forwarding**. In this case, you connect a port from a remote server to another server (mostly your own computer). Using this, you can expose a webserver from your own computer on the Internet:

Copy

```
$ ssh -R 8000:localhost:80 root@5.161.197.79
```

In the above example, I'm telling the ssh to create a **Remote** tunnel. After this any request on port 8000 from the 5.161.197.79 computer will reach to localhost (my machine) port 80!

You can even ask the remote machine to start listening on a specific port on all of its interfaces (open up to the network and not only localhost):

Copy

```
ssh -R 0.0.0.0:8000:localhost:7777 192.168.70.2
```

Above, I'm telling the 192.168.70.2 machine to open port 8000 to ALL interfaces and forward whatever it got to my machiens 7777 port.

> to let clients to *bind* listening ports to anything other than localhost, the `GatewayPorts clientspecified` configu should be set in the ssh server's config file.

There are even more use caases. For example you use the -D switch for a `dynamic` application level port forarding. It works like a proxy / anti censorship. If I do

Copy

```
ssh -D 1080 192.168.70.2
```

The 1080 port will work ask a *socks* proxy on my machine and forward whatever request reaches it to the 192.168.70.2 machine and returns back the answers. Now I can configure my applications to use the localhost:1080 as their socks proxy and the 192.168.70.2 will work as a proxy here. This is useful when you do not have internet on your localhost, but 192.168.70.2 has it.

X Forwarding

The last concept I want to talk about in forwarding section, is the X forwarding. As you already know from Module 106, linux graphical applications use X as their graphical host. This X can also be forwarded on the network as any other network based communication. Its even has its own switch on the ssh program. To forward the X, its enough to add a `-X` to your ssh. Please note that the `X11Forwarding yes` configuration should be present in your sshd_config file.

To activate the X11 forwarding in a ssh session, simply add `-X` to your ssh:

Copy

```
ssh -X 192.168.70.2
```

and then you can run graphical programs (say `xeyes` for fun) on the remote machine and see the graphical part on your own X server!

> If you are not on a GNU/Linux machine with an X11 server, you need to install it (for example XQuartz on Mac)

This is very useful when you want to run a program on a resourceful server or work from home whiel your graphical applications are at the office ;)

ecnrypt and sign using gpg

As described in previous section, a public and private key pair can be used to encrypt or sign messages. There is an implementation of this method called `gpg` which can be used on Linux (and other machiens) to perform these tasks. First you need to generate a key:

Copy

```
jadi@debian:~$ gpg --gen-key
gpg (GnuPG) 2.2.40; Copyright (C) 2022 g10 Code GmbH
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.

gpg: directory '/home/jadi/.gnupg' created
gpg: keybox '/home/jadi/.gnupg/pubring.kbx' created
Note: Use "gpg --full-generate-key" for a full featured key generation dialog.

GnuPG needs to construct a user ID to identify your key.

Real name: Jadi
Name must be at least 5 characters long
Real name: Jadi M
Email address: jadijadi@gmail.com
You selected this USER-ID:
    "Jadi M <jadijadi@gmail.com>"

Change (N)ame, (E)mail, or (O)kay/(Q)uit?
Change (N)ame, (E)mail, or (O)kay/(Q)uit? O
We need to generate a lot of random bytes. It is a good idea to perform
some other action (type on the keyboard, move the mouse, utilize the
disks) during the prime generation; this gives the random number
generator a better chance to gain enough entropy.
We need to generate a lot of random bytes. It is a good idea to perform
```

```
some other action (type on the keyboard, move the mouse, utilize the
disks) during the prime generation; this gives the random number
generator a better chance to gain enough entropy.
gpg: /home/jadi/.gnupg/trustdb.gpg: trustdb created
gpg: directory '/home/jadi/.gnupg/openpgp-revocs.d' created


gpg: revocation certificate stored as '/home/jadi/.gnupg/openpgp-revocs.d/E279681DC09318AB5B23D359C9063DD26365986A.rev'
public and secret key created and signed.

pub   rsa3072 2023-09-17 [SC] [expires: 2025-09-16]
      E279681DC09318AB5B23D359C9063DD26365986A
uid                      Jadi M <jadijadi@gmail.com>
sub   rsa3072 2023-09-17 [E] [expires: 2025-09-16]
```

Now the key is created in the `~/.gnupg` directory and we can share our public key with others:

Copy

```
jadi@debian:~$ gpg --list-keys
/home/jadi/.gnupg/pubring.kbx
-----------------------------
pub   rsa3072 2023-09-17 [SC] [expires: 2025-09-16]
      E279681DC09318AB5B23D359C9063DD26365986A
uid           [ultimate] Jadi M <jadijadi@gmail.com>
sub   rsa3072 2023-09-17 [E] [expires: 2025-09-16]

jadi@debian:~$ gpg --export jadi > jadi.pub.key
jadi@debian:~$ file jadi.pub.key
jadi.pub.key: OpenPGP Public Key Version 4, Created Sun Sep 17 12:39:19 2023, RSA (Encrypt or Sign, 3072 bits); User ID; Signature; OpenPGP Ce
```

> the exported key is in binary format, add `-a` to armor it in ascii

Now you have to distribute this key to others. You can upload it somewhere, email it, put it on your website or use some of the public key stores to share it with other. If anyone recives it, she can import it into her key store using the following command:

Copy

```
gpg --import jadi.pub.key
```

Oh! and what happens if you feel that your key is compromised!? In this case you have to create a **revoke** file and share it with other to tell them that your key is not valid anymore:

Copy

```
gpg --output jadi.revoke.asc --gen-revoke jadijadi@gmail.com
```

This tells gpg to create a revoke file called jadi.revoke.asc for the identity jadijadi@gmail.com. If jadijadi@gmail.com needs to invalidate his public key, he has to publish this file to the Internet or key servers and others will know that the previous public key is not valid anymore.

Encrypt / Decrypt files

At this stage, I have a key and my friends do have my public key imported in their machine. If any of them wants to send my an encrypted message, they can do the following:

Copy

```
echo "I Loved your course! I'll tell all my friends about it." > file.txt
gpg --out file.txt.encrypted --recipient jadijdai@gmail.com --encrypt file.txt
```

Now its enough for them to send me this file. Even using unsecure channels because its encrypted by the top notch tools. Every one can download this file but ONLY I will be able to decypt it; because only I have the **Private key** of jadijadi@gmail.com.

Copy

```
gpg --out out.txt --decrypt file.txt.encrypted
```

> If you liked this course, no need to encrypt the message, just share the linux1st.com on the internet!

Signing and verifying files

In the previous section, we used gpg to encrypt the data. We used someone's Public key to encrypt and then used our own Private Key to decrypt the data. But what happens if I use my Private key on my side and let other use my Public key to *open* the data? That is called **signing*.

Please remember that only I have access to my private key. So if I apply it to a file, every one:

1. will be able to open the file using my Public Key
2. Will be sure that **I** have signed this because only **I** has access to the private key of the public key they used to open the file.

Lets sign a message:

Copy

```
$ echo "I'm Jadi and I'm glad that you reached to the end of your LPIC study" > message-jadi.txt
$ gpg --output message-jadi.sig --sign message-jadi.txt
```

Now its enough for me to publish the `message-jadi.sig` to the Internet or send it to someone. If they want to make sure that it is truely comming from me, its enough for them to check my signature (obviously after importing my public key):

Copy

```
jadi@debian:~$ gpg --verify message-jadi.sig
gpg: Signature made Sun 17 Sep 2023 09:26:15 AM EDT
gpg:                using RSA key E279681DC09318AB5B23D359C9063DD26365986A
gpg: Good signature from "Jadi M <jadijadi@gmail.com>" [ultimate]
```

This only checked the signature, if they also needed to decrypt the message they had to as follow:

Copy

```
jadi@debian:~$ gpg --output message.jadi --decrypt message-jadi.sig
gpg: Signature made Sun 17 Sep 2023 09:26:15 AM EDT
gpg:                using RSA key E279681DC09318AB5B23D359C9063DD26365986A
gpg: Good signature from "Jadi M <jadijadi@gmail.com>" [ultimate]
jadi@debian:~$ cat message.jadi
I'm Jadi and I'm glad that you reached to the end of your LPIC study
```

Please also note the `--clearsign` option. This option will create a file ending in `.asc` which containt the original un-encrypted (clear text) message alongside the signature. This way non-tech-savy people will be able to read the message too and only if someone wants, she can `--verify` the signature.

Copy

```
jadi@debian:~$ cat message-jadi.txt
I'm Jadi and I'm glad that you reached to the end of your LPIC study
jadi@debian:~$ gpg --clearsign message-jadi.txt
jadi@debian:~$ ls -ltrh
total 16M
drwxr-xr-x 27 jadi jadi 4.0K Jun 15 09:00 rust-for-linux
drwxr-xr-x  2 jadi jadi 4.0K Jun 15 09:09 Downloads
drwx------  2 jadi jadi 4.0K Jul 17 10:53 BRF
-rw-r--r--  1 jadi jadi    6 Jul 17 10:57 myfile
drwxr-xr-x  5 jadi jadi 4.0K Jul 25 13:34 nltk_data
drwxr-xr-x  4 jadi jadi 4.0K Aug 13 07:29 w
-rw-r--r--  1 jadi jadi  16M Aug 24 01:15 nvim-linux64.tar.gz
drwxr-xr-x  6 jadi jadi 4.0K Aug 24 05:38 nvim-linux64
-rw-r--r--  1 jadi jadi 3.8K Aug 27 15:49 report.xml
-rw-r--r--  1 jadi jadi 1.8K Sep 17 08:50 jadi.pub.key
-rw-r--r--  1 jadi jadi   69 Sep 17 09:26 message-jadi.txt
-rw-r--r--  1 jadi jadi  549 Sep 17 09:26 message-jadi.sig
-rw-r--r--  1 jadi jadi   69 Sep 17 09:27 message.jadi
-rw-r--r--  1 jadi jadi  777 Sep 17 09:28 message-jadi.txt.asc
jadi@debian:~$ cat message-jadi.txt.asc
-----BEGIN PGP SIGNED MESSAGE-----
Hash: SHA512

I'm Jadi and I'm glad that you reached to the end of your LPIC study
-----BEGIN PGP SIGNATURE-----

iQGzBAEBCgAdFiEE4nloHcCTGKtbI9NZyQY90mNlmGoFAmUG/xYACgkQyQY90mNl
mGqpXQv/QNv0NlCR+S7HJhxiOZcKXE3RB1sx4jRVrOtMd6ZoXYLb28Zz4xN/Lzh2
sipohDbNttr1ovNZvMMu83bbuSeViY9L7nvjNjKsY50u5V1KnrMJPHRaGV8g/0p2
t5zFJuzSFWXQczIjjXMpoLVH8jK3mJuBOmVFwLuBqUoW8827tCLIdr1DNyHXBW47
fcLK5/bO/ljdwGnAX46d+Prg0TIfMLjPpb7uFXVkRBo+zMtCEVDGObaVHQ3J8t6w
8i2YgMrmA3h9h4Gy/yM9Agb2kPeKl6iToeCfQ0EwTcnPF1TN0KZ1PS8Pr3Jp3LcS
A+p32zj7YTzTeE1uBAb7/BorBqaArX8HtLe15yFkNKhPluXya9GG4/Nsxk33Qpd3
c1KiLK9YhTFy7u+Q6tbTKG+uuGj71wANhkqRP30iMjLcBQ7aAO3ehVSYZFyZ7dbL
SJollwyG1sf4zH0g5SQZOxpjyz874IrlFc7GldDCr5jXZw6H+4NuY2yiVWKAcOcU
Qv23xWL2
=q81y
-----END PGP SIGNATURE-----
jadi@debian:~$ gpg --verify message-jadi.txt.asc
gpg: Signature made Sun 17 Sep 2023 09:28:54 AM EDT
gpg:                using RSA key E279681DC09318AB5B23D359C9063DD26365986A
gpg: Good signature from "Jadi M <jadijadi@gmail.com>" [ultimate]
gpg: WARNING: not a detached signature; file 'message-jadi.txt' was NOT verified!
```

here the --clearsign tells the gpg to include the clear text message in the output file too. The output file will be originalfile.asc

and another one to verify that a document is singed correctly:

Copy

```
gpg --verify recievedfile
```

gpg-agent

Just like the `ssh-agent`, `gpg-agent` is a tool which acts like a password manager for you gpg keys. It keeps the keys in the memory so you wont need to provide password on every single use.

Powered by: Pelican Theme: Elegant